10/541029
PCT/JP2004/009043

21.06.2004

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２００３年　６月３０日

出願番号
Application Number: 特願２００３－１８７６９０

[ST. 10/C]: [ＪＰ２００３－１８７６９０]

出願人
Applicant(s): 松下電器産業株式会社

２００４年　７月２３日

特許庁長官
Commissioner,
Japan Patent Office

小川洋

出証番号　出証特２００４－３０６４６５４

【書類名】　特許願
【整理番号】　2968250008
【提出日】　平成15年 6月30日
【あて先】　特許庁長官　殿
【国際特許分類】　G05B　9/06
【発明者】
　【住所又は居所】　東広島市鏡山３丁目１０番１８号　株式会社松下電器情報システム広島研究所内
　【氏名】　今西　祐子
【発明者】
　【住所又は居所】　東広島市鏡山３丁目１０番１８号　株式会社松下電器情報システム広島研究所内
　【氏名】　土井　繁則
【特許出願人】
　【識別番号】　000005821
　【氏名又は名称】　松下電器産業株式会社
【代理人】
　【識別番号】　100090446
　【弁理士】
　【氏名又は名称】　中島　司朗
【手数料の表示】
　【予納台帳番号】　014823
　【納付金額】　21,000円
【提出物件の目録】
　【物件名】　明細書　1
　【物件名】　図面　1
　【物件名】　要約書　1
　【包括委任状番号】　9003742
【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　ガーベジコレクションシステム

【特許請求の範囲】

【請求項１】　複数スレッドで構成されるオブジェクト指向プログラムの実行過程において不要になったオブジェクトに対応するメモリ領域を解放するガーベジコレクションシステムであって、

複数スレッドそれぞれを順に選択する選択手段と、

選択されたスレッドについて、当該スレッドの実行を停止して、当該スレッドからオブジェクトポインタの参照を通じてアクセス可能であるオブジェクトを検出して当該検出したオブジェクトを非解放対象として管理して、当該スレッドの実行を再開するという手順からなる検査処理を実施する検査手段と、

前記選択手段による前記選択が開始された後において、実行中のスレッドによりオブジェクトポインタが処理対象にされたことを検知した場合には、当該オブジェクトポインタが指すオブジェクトを非解放対象として管理する検知手段と、

前記複数スレッド全てについて前記検査処理が完了した後において、非解放対象として管理されているオブジェクト以外のオブジェクトに対応するメモリ領域を、解放する解放手段とを備える

ことを特徴とするガーベジコレクションシステム。

【請求項２】　前記検知手段は、前記検知を、実行中のスレッドについて未だ検査処理がなされていない場合に限り行い、

前記検知手段は、

前記検知をした場合に、当該実行中のスレッドに処理対象にされたオブジェクトポインタ、及び当該オブジェクトポインタから辿ることのできるオブジェクト内のオブジェクトポインタを、当該スレッドに対応する作業用メモリ領域に保存する検出部と、

前記検査手段によりスレッドの実行が停止されている間に、当該スレッドに対応する作業用メモリ領域内のオブジェクトポインタから辿ることのできるオブジェクトを非解放対象として管理する管理部とを有する

ことを特徴とする請求項１記載のガーベジコレクションシステム。

【請求項３】　前記検査処理は、

選択された当該スレッドに対応するスタック内のオブジェクトポインタが指すオブジェクトを、前記アクセス可能であるとして検出した際において、

検出した当該オブジェクトが既に非解放対象として管理されておらず、かつ、当該オブジェクト内にオブジェクトポインタがあるときに限り、当該オブジェクトポインタが指すオブジェクトを更に前記アクセス可能であるとして検出する手順を、

繰り返し行うことを内容とする処理であり、

前記選択手段は、最初の選択後は、前記検査手段により前記検査処理が行われた後において前記複数スレッドのうち前記検査処理を実施されていないスレッドがある限り更なる選択を行い、

前記選択手段は、各スレッドに関する情報を参照し、所定のスレッド選定条件に基づき、前記選択を行う

ことを特徴とする請求項２記載のガーベジコレクションシステム。

【請求項４】　前記スレッド選定条件は、スレッド状態がｗａｉｔ状態であるスレッドをスレッド状態がｗａｉｔ状態以外であるスレッドよりも早期に選択することを示す条件を含み、

前記選択手段は、前記選択を行う時点でｗａｉｔ状態のスレッドがある限り、当該ｗａｉｔ状態のスレッドを選択する

ことを特徴とする請求項３記載のガーベジコレクションシステム。

【請求項５】　前記スレッド選定条件は、スレッド優先度のスレッド優先度の低いスレッドをスレッド優先度の高いスレッドよりも早期に選択することを示す条件を含む

ことを特徴とする請求項３又は請求項４記載のガーベジコレクションシステム。

【請求項６】　前記スレッド選定条件は、スレッドに対応するスタックサイズが小さいスレッドをスタックサイズが大きいスレッドよりも早期に選択することを示す条件を含む

ことを特徴とする請求項３～５のいずれか１項に記載のガーベジコレクション

システム。

【請求項７】　コンピュータ上において複数スレッドで構成されるオブジェクト指向プログラムの実行過程において不要になったオブジェクトに対応するメモリ領域を解放するガーベジコレクション方法であって、

複数スレッドそれぞれを順に選択する選択ステップと、

選択されたスレッドについて、当該スレッドの実行を停止して、当該スレッドからオブジェクトポインタの参照を通じてアクセス可能であるオブジェクトを検出して当該検出したオブジェクトを非解放対象として管理して、当該スレッドの実行を再開するという手順からなる検査処理を実施する検査ステップと、

前記選択ステップによる前記選択が開始された後において、実行中のスレッドによりオブジェクトポインタが処理対象にされたことを検知した場合には、当該オブジェクトポインタが指すオブジェクトを非解放対象として管理する検知ステップと、

前記複数スレッド全てについて前記検査処理が完了した後において、非解放対象として管理されているオブジェクト以外のオブジェクトに対応するメモリ領域を、解放する解放ステップとを含む

ことを特徴とするガーベジコレクション方法。

【請求項８】　複数スレッドで構成されるオブジェクト指向プログラムの実行過程において不要になったオブジェクトに対応するメモリ領域を解放するガーベジコレクション処理をコンピュータに実行させるためのコンピュータプログラムであって、

前記ガーベジコレクション処理は、

複数スレッドそれぞれを順に選択する選択ステップと、

選択されたスレッドについて、当該スレッドの実行を停止して、当該スレッドからオブジェクトポインタの参照を通じてアクセス可能であるオブジェクトを検出して当該検出したオブジェクトを非解放対象として管理して、当該スレッドの実行を再開するという手順からなる検査処理を実施する検査ステップと、

前記選択ステップによる前記選択が開始された後において、実行中のスレッドによりオブジェクトポインタが処理対象にされたことを検知した場合には、当該

オブジェクトポインタが指すオブジェクトを非解放対象として管理する検知ステップと、

前記複数スレッド全てについて前記検査処理が完了した後において、非解放対象として管理されているオブジェクト以外のオブジェクトに対応するメモリ領域を、解放する解放ステップとを含む

ことを特徴とするコンピュータプログラム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、アプリケーションプログラム（ＡＰ：Application Program）が使用したメモリについてのガーベジコレクション（ＧＣ：Garbage Collection）に関する。

【０００２】

【従来の技術】

従来のオブジェクト指向のプログラミング言語には、ＡＰの作成者にメモリ領域の確保や解放を意識させず、ＡＰが使用した後に不要となったオブジェクト（オブジェクトインスタンス）に対応するメモリ領域の解放を実行環境側に任せる方法を採るものがある。例えばＪａｖａ（登録商標）言語である。なお、Ｊａｖａ（登録商標）は米国Ｓｕｎ　Ｍｉｃｒｏｓｙｓｔｅｍｓ社の商標である。

【０００３】

このような言語を用いて記述されたＡＰは、不要となったオブジェクトに対応するメモリ領域を自動的に解放するガーベジコレクション（ＧＣ）機構を備える実行環境上で動かされる。

ＧＣ機構は、ＡＰの実行に際して動的に確保されたオブジェクトに対応するメモリ領域がどこからも参照されていない状態になっていることを検出してそのメモリ領域を解放し再利用可能状態にする。

【０００４】

マルチスレッド構成のＡＰにおいて、各スレッドは、それぞれスタック領域と対応し、動作過程において、スタック領域内にデータを格納したり、格納したデ

ータを参照したり、オブジェクトを生成したりする。オブジェクトを生成した場合には通常はスタック領域内にそのオブジェクトのポインタ、つまりそのオブジェクトのメモリ内における所在位置を指し示すデータ（以下、「オブジェクトポインタ」という。）が格納され、スレッドからのオブジェクトへのアクセスはそのオブジェクトポインタを参照することによりなされる。また、通常、オブジェクト内領域にも、他のオブジェクトへのオブジェクトポインタが格納される。

【０００５】

ある時点においてＡＰにより参照されている全てのオブジェクトは、ＡＰの各スレッドに対応する各スタック領域内に含まれるオブジェクトポインタから、直接、又は１以上のオブジェクト内のオブジェクトポインタを媒介にして、辿ることができる。

これに対応してＧＣ機構は、基本的に、ある時点においてスタック領域内のオブジェクトポインタから辿れなくなっているオブジェクトに対応するメモリ領域を不要なものとして解放対象にする。

【０００６】

従来のＧＣ方式として知られているマークアンドスイープ方式は、特定のオブジェクトポインタから辿れる全てのオブジェクトにマークを付けていく処理を行った後に、全てのオブジェクトを走査して、マークの付いていないオブジェクトに対応するメモリ領域を解放する方式である。

このマークアンドスイープ方式のＧＣをマルチスレッド構成のＡＰの実行時に行う場合に、ＧＣの迅速性を最優先して単純に、全スレッドを停止してから、各スレッドに対応するスタック領域内のオブジェクトポインタから辿れる全てのオブジェクトにマークを付与する処理を行い、その後に全スレッドの停止を解除して、マークの付いていないオブジェクトの解放を行うとするならば、次の問題が生じる。

【０００７】

即ち、ＡＰの全スレッドが停止している期間が長期化する可能性があり、この場合に、ユーザの操作等に対して一切反応を示せず、例えばコンピュータのディスプレイの表示内容は変化しないままとなり、ユーザを戸惑わせてしまうという

問題である。

この問題を解決する方式として、このマークアンドスイープを用いたＧＣを、マルチスレッド構成のＡＰを全く停止することなく実行する方式が提案されている（特許文献１参照）。

【０００８】

この方式は、ルートノードなるオブジェクトから辿れる全てのオブジェクト及びスレッド毎の各スタック領域内のオブジェクトポインタから辿れる全てのオブジェクトに対してマークを付与する第１処理を行い、その第１処理中に、ＡＰのスレッド（以下、「ＡＰスレッド」という。）の動作によりオブジェクトへのオブジェクトポインタが移動した場合にそのオブジェクトを表すデータをマークスタックなる領域に積んでおき、そのマークを付与する処理が完了した段階で、更にマークスタックから辿れる全てのオブジェクトに対してマークを付与する第２処理を行い、最後に、マークされていないオブジェクトに対応するメモリ領域を解放するものである。

【０００９】

【特許文献１】

特許第３０２７８４５号公報

【００１０】

【発明が解決しようとする課題】

しかしながら、上述のＡＰを全く停止せずにマークを付与する方式では、ＡＰスレッドの動作によってスタック領域内のデータが変化するため、第１処理中、スタック領域内のオブジェクトポインタから辿れるオブジェクトに対してマークを付与する処理の一部が無駄となる可能性がある。

【００１１】

例えば、ＧＣを行うスレッドが、ＡＰの１つのスタック領域内のオブジェクトポインタ（オブジェクトポインタＡ）を検出して、そのオブジェクトポインタＡから辿れるオブジェクトに対してマークを付与する処理Ａを実行している間に、そのスタック領域に対応するＡＰスレッドが、オブジェクトポインタＡ或いはそれらのオブジェクト内の１又は複数のオブジェクトポインタを、コピーしてスタ

ック領域に新たに格納する動作をしたならば、ＧＣを行うスレッドはその処理Ａの終了後にいずれ、再び処理Ａと一部重複する処理を行う、或いはその重複防止のためのチェック処理をわざわざ行うことになり、これが無駄となる。この処理の無駄は、ＧＣの開始から完了までに要するＣＰＵ時間を無駄に増加させることにつながり、ＣＰＵの利用効率を低下させる。

【００１２】

そこで、本発明は、上記問題に鑑みてなされたものであり、ＡＰの全スレッドが停止している期間を長期化させないようにするとともに、ＧＣの開始から完了までに要するＣＰＵ時間の無駄な増加をある程度抑制するようなＧＣ方式を用いるガーベジコレクション（ＧＣ）システムを提供することを目的とする。

【００１３】

【課題を解決するための手段】

上記課題を解決するために、本発明に係るＧＣシステムは、複数スレッドで構成されるオブジェクト指向プログラムの実行過程において不要になったオブジェクトに対応するメモリ領域を解放するガーベジコレクションシステムであって、複数スレッドそれぞれを順に選択する選択手段と、選択されたスレッドについて、当該スレッドの実行を停止して、当該スレッドからオブジェクトポインタの参照を通じてアクセス可能であるオブジェクトを検出して当該検出したオブジェクトを非解放対象として管理して、当該スレッドの実行を再開するという手順からなる検査処理を実施する検査手段と、前記選択手段による前記選択が開始された後において、実行中のスレッドによりオブジェクトポインタが処理対象にされたことを検知した場合には、当該オブジェクトポインタが指すオブジェクトを非解放対象として管理する検知手段と、前記複数スレッド全てについて前記検査処理が完了した後において、非解放対象として管理されているオブジェクト以外のオブジェクトに対応するメモリ領域を、解放する解放手段とを備えることを特徴とする。

【００１４】

上記構成により、ＡＰスレッドからスタック内やオブジェクト内のオブジェクトポインタを介して辿ることができるオブジェクトについての非解放対象として

の指定の過程つまりマーク付与の過程においては、そのＡＰスレッドを停止させているため、そのＡＰスレッドが動作することに起因してスタック領域内のデータが変化するという事態が起こらないため、その過程におけるマーク付与が無駄になる危険性はなくなり、この点についてはＣＰＵの利用効率低下を防止できる。また、これにより、全てのＡＰスレッドを停止して上述のマーク付与を行っているのではないため、ＡＰの全スレッドが停止している期間の長期化が防止できる。

【００１５】

【発明の実施の形態】

＜実施形態１＞

以下、本発明の実施形態１に係るガーベジコレクション（ＧＣ）システムについて図面を用いて説明する。

＜構成＞

図１は、本発明の実施形態１に係るＧＣシステムの機能ブロック図である。

【００１６】

ＧＣシステム１０は、ＣＰＵ、メモリ等を備えたコンピュータにおいてメモリに格納された制御プログラムがＣＰＵによって実行されることにより実現され、マルチスレッド制御を行う一般のオペレーティングシステム（ＯＳ：Operating System）及びいわゆる仮想マシンを含み、Ｊａｖａ（登録商標）言語等で作成されたアプリケーションプログラムの実行環境として位置づけられるシステムである。

【００１７】

ＧＣシステム１０は、図１に示すように、インタプリタ部１００、オブジェクト管理部２００、スレッド管理部３００及びＧＣ部４００を備える。

ここで、インタプリタ部１００は、基本的にインタプリタであってＡＰを実行する機能を担い、命令実行部１１０及びオブジェクト参照検出部１２０を有する。

【００１８】

オブジェクト管理部２００は、オブジェクトを管理する機能を担い、ｆｒｏｍ

テーブル２２１、ｔｏテーブル２２２、オブジェクト等に対応するメモリ領域であるオブジェクト情報記憶部２１０、オブジェクト生成部２３０及びテーブル切替部２４０を有する。

なお、ｆｒｏｍテーブル２２１は、ＧＣ開始前において存在する全オブジェクトに対する全オブジェクトポインタを格納するものと位置づけられるテーブルであり、ｔｏテーブル２２２は、マークアンドスイープ方式でのマーク付与がなされたオブジェクトに対するオブジェクトポインタを格納するために利用されるテーブルである。これらのテーブルについては後に詳細に説明する。

【００１９】

スレッド管理部３００は、マルチスレッド制御を実現し、スレッドを管理する機能を担うものであり、スレッド制御部３１０と、スレッド毎に、スレッド情報３２０、オブジェクト参照情報３３０及びスタック３４０に対応するメモリ領域を有する。

なお、スレッド情報３２０は、ＧＣ処理全体の開始時点でＯＮにされそのスレッドに対しての参照処理が終了するとＯＦＦにされるＧＣフラグを含む。参照処理は、オブジェクトへのマーク付与に相当する処理であり、オブジェクトを参照するためにスタック内等に格納されているオブジェクトポインタと同一内容のオブジェクトポインタをｆｒｏｍテーブルからｔｏテーブルに移動することを主内容とする処理である。

【００２０】

スレッド管理部３００は、各スレッドを並列的に実行する。即ち、スレッド管理部３００は、例えば微小時間毎に各スレッドを切り替えて実行する、或いは各スレッドをマルチプロセッサにおける複数のプロセッサエレメントによって並列実行する。

ＧＣ部４００は、基本的にいわゆるガーベジコレクタに相当し、ＧＣ制御部４１０、スレッド選定条件記憶部４２０、参照処理部４３０及び解放部４４０を有する。このＧＣ部４００を中心として行われるＧＣは、基本的にマークアンドスイープ方式を用い、参照処理の対象となるＡＰスレッドを順番に選択して、選択したスレッドを停止させて参照処理を行ってからその停止を解除する方式のもの

である。

【００２１】

インタプリタ部１００における命令実行部１１０は、ＡＰスレッドを構成する命令列を逐次解釈して実行し、命令がオブジェクトの生成命令である場合にはオブジェクト生成部２３０にオブジェクトを生成させる機能を有する。

オブジェクト参照検出部１２０は、ＧＣフラグがＯＮである場合に限って、命令実行部１１０により実行されるスレッド内の命令が、オブジェクトのオブジェクトポインタを処理対象とする命令であるときに、そのオブジェクトポインタを、オブジェクト参照情報３３０のメモリ領域の中に格納する機能を有する。

【００２２】

オブジェクト管理部２００におけるオブジェクト生成部２３０は、いわゆるクラスファイル等のオブジェクトの定義情報を参照して、ＯＳのメモリ管理機構によりオブジェクトに対応するメモリ領域を確保することにより、メモリ内にオブジェクトを生成する機能を有する。

テーブル切替部２４０は、ｆｒｏｍテーブルを指すものとして用いられているポインタとｔｏテーブルを指すものとして用いられているポインタとを交換することにより、瞬時にｆｒｏｍテーブルの内容とｔｏテーブルの内容とを交換したに等しい効果を生じさせる機能を有する。

【００２３】

スレッド管理部３００におけるスレッド制御部３１０は、マルチスレッド制御を行い、ＡＰを構成する各スレッド及びＧＣ処理のスレッドを擬似並行的に実行する。

ＧＣ部４００におけるスレッド選定条件記憶部４２０は、参照処理を行う対象となるスレッドを選択するための条件を示すスレッド選定条件を記憶しているメモリ領域である。

【００２４】

参照処理部４３０は、参照処理を行う機能を有し、解放部４４０は、マーク付与がなされていないオブジェクトの解放、即ち、ＧＣの終了段階においてｆｒｏｍテーブル２２１に残存しているオブジェクトポインタが指すオブジェクトに対

応するメモリ領域を解放する機能を有する。

また、ＧＣ制御部４１０は、ＧＣ制御処理を実行する機能を有する。即ち、ＧＣ制御部４１０は、スレッド選定条件を参照することにより、参照処理の対象にするＡＰスレッドを１つ選定し、選択したスレッドについて、スレッド制御部３１０にそのスレッドを停止させてから、参照処理部４３０にそのスレッドに対応するスタック及びオブジェクト参照情報に基づいて参照処理を行わせ、その後にスレッド制御部３１０に選択したスレッドの停止を解除させてから、次のＡＰスレッドの選択を行うという手順を、未処理のＡＰスレッドが存在しなくなるまで繰り返してから、解放部４４０に、マーク付与がなされていないオブジェクトの解放を行わせる機能を有する。

【００２５】

＜データ＞

以下、ＧＣシステム１０において取り扱われるデータについて説明する。

図２は、オブジェクトとオブジェクトポインタとの関係を例示する図である。

オブジェクトを指すつまりオブジェクトの所在位置を指すオブジェクトポインタは、オブジェクト内、スタック内、又は共有オブジェクト管理情報３５３内に存在し得る。ここで、共有オブジェクト管理情報３５３は、ＡＰの実行環境としての仮想マシンにおいて必要なため確保されているオブジェクト群に対するオブジェクトポインタ群を含むデータである。

【００２６】

図２の例では、共有オブジェクト管理情報３５３内のあるオブジェクトポインタがオブジェクト２０１ｃを指している。

また、スレッド３５１に対応するスタック内のあるオブジェクトポインタは、オブジェクト２０１ａを指しており、つまり、オブジェクト２０１ａのメモリアドレスを内容としており、また別のオブジェクトポインタは、オブジェクト２０１ｄを指している。従って、スレッド３５１は、実行中にこれらのオブジェクトポインタを参照することによりオブジェクト２０１ａ、オブジェクト２０１ｄにアクセス可能な状態である。

【００２７】

また、スレッド３５２に対応するスタック内のあるオブジェクトポインタは、オブジェクト２０１ｂを指しており、オブジェクト２０１ｂ内のデータとして、オブジェクト２０１ｅを指すオブジェクトポインタが含まれている。従って、スレッド３５２は、これらのオブジェクトポインタを辿ることによりオブジェクト２０１ｅにアクセス可能な状態である。

【００２８】

図３は、ｆｒｏｍテーブル及びｔｏテーブルを示す図である。

オブジェクト情報記憶部２１０内に、オブジェクトポインタ又はｎｕｌｌ値を十分な数だけ格納するための２つのテーブルが設けられており、また、ｆｒｏｍテーブルポインタ２１１及びｔｏテーブルポインタ２１２が存在する。ｆｒｏｍテーブルポインタ２１１は、その２つのテーブルのうち一方を指すポインタであり、ｔｏテーブルポインタ２１２は、他方を指すポインタである。ここでは、現時点でｆｒｏｍテーブルポインタ２１１が指しているテーブルをｆｒｏｍテーブルと称しており、ｔｏテーブルポインタ２１２が指しているテーブルをｔｏテーブルと称している。

【００２９】

ｆｒｏｍテーブル２２１は、ＧＣ開始時点においては、その時に存在する全オブジェクトに対する全オブジェクトポインタを格納している。

また、ｔｏテーブル２２２は、参照処理の過程でｆｒｏｍテーブル２２１内から移動してオブジェクトポインタを格納するテーブルである。なお、ｆｒｏｍテーブル２２１中のオブジェクトポインタがｔｏテーブル２２２へと移動されるときには、そのオブジェクトポインタが格納されていたｆｒｏｍテーブル２２１内の位置にあたるメモリ内容はｎｕｌｌ値にされる。

【００３０】

図４は、参照処理によるｆｒｏｍテーブル及びｔｏテーブルの内容の変化を示す図であり、図４（ａ）は参照処理開始前の状態を示し、図４（ｂ）は参照処理後の状態を示す。

図４（ａ）では、参照処理開始前におけるｆｒｏｍテーブル２２１ｘは、ｏｂｊＡ２０２ａ、ｏｂｊＢ２０２ｂ、ｏｂｊＣ２０２ｃのそれぞれのオブジェクト

を指すオブジェクトポインタを含んでいる。

【００３１】

その後にＡＰスレッド３５４を対象として参照処理が行われた場合には、ＡＰスレッド３５４のスタック内に含まれているオブジェクトポインタと同値のものがｆｒｏｍテーブルからｔｏテーブルに移動され、その結果として図４（ｂ）に示す状態になる。ｆｒｏｍテーブル２２１ｙでは、ｏｂｊＡ２０２ａを指していたオブジェクトポインタの存在した場所と、ｏｂｊＣ２０２ｃを指していたオブジェクトポインタの存在した場所との両方がｎｕｌｌ値に更新されており、ｔｏテーブル２２２ｙには、ｏｂｊＡ２０２ａを指すオブジェクトポインタと、ｏｂｊＣ２０２ｃを指すオブジェクトポインタとが格納されている。

【００３２】

図５は、スレッド情報、スタック及びオブジェクト参照情報を示す図である。

スレッド情報３２０は、スレッド毎に、スレッド生成時に生成され、そのスレッドについての情報を含むものであり、具体的には、状態３２１、優先度３２２、スタックポインタ３２３、ＧＣ開始時スタックポインタ３２４、ＧＣフラグ３２５、オブジェクト参照情報先頭ポインタ３２６及びオブジェクト参照情報カレントポインタ３２７を含む。なお、スレッド生成時には、スレッド情報３２０のメモリ領域のみならずスタック３４０のメモリ領域も確保される。

【００３３】

スレッド情報３２０における状態３２１は、マルチスレッド制御のためのスレッド状態を示す情報であり、ｗａｉｔ状態、ｒｕｎ状態、ｒｅａｄｙ状態等の別を示す。

優先度３２２は、スレッドの優先度を示す情報であり、優先度は例えばスレッド生成時にＡＰからの指定を受けて定まる。なお、マルチスレッド制御において優先度が高いスレッドがｒｅａｄｙ状態であればそれより優先度の低いスレッドよりも優先的に実行対象とされる。

【００３４】

スタックポインタ３２３は、該当のスレッドについてのスタック内における現在の有効なデータ範囲の終端を示すものである。なお、マルチスレッド制御にお

いて、スレッドがｒｕｎ状態から、他の状態つまり停止した状態に切り替えられる際に、それまでスタックポインタを指す所定レジスタに格納されていた値がこのスタックポインタ３２３に格納され、スレッドがｒｕｎ状態に切り替えられる際に、このスタックポインタ３２３内の値がその所定レジスタに設定される。

【００３５】

ＧＣ開始時スタックポインタ３２４は、ＧＣ開始時におけるスタック内における有効なデータ範囲の終端を示すものである。

ＧＣフラグ３２５は、ＧＣ処理全体の開始時点でＯＮにされ、そのスレッドに対しての参照処理が終了するとＯＦＦにされる。

オブジェクト参照情報先頭ポインタ３２６は、該当スレッドに対応するオブジェクト参照情報のメモリ領域の先頭を示すものであり、スレッド生成の際のそのメモリ領域確保時に設定される。

【００３６】

また、オブジェクト参照情報カレントポインタ３２７は、該当スレッドに対応するオブジェクト参照情報のメモリ領域において、オブジェクト参照検出部１２０が次にオブジェクトポインタを格納すべき位置を示す情報であり、オブジェクト参照検出部１２０よって参照及び更新される。

図６は、参照処理とＡＰスレッドの状態との関係を示す図である。

【００３７】

ＧＣスレッド３５６は、ＧＣ制御部４１０によるＧＣを実行するスレッドであり、ＧＣとして各ＡＰスレッドを順番に対象として参照処理を行う。ここでは、ＡＰスレッド３５５ａ、ＡＰスレッド３５５ｂ、ＡＰスレッド３５５ｃの順に参照処理が行われる場合の例を示している。

ＡＰスレッド３５５ａは、参照処理がなされた後の状態であり、実行中である。ここで実行中とは、ｓｌｅｅｐ状態つまり停止状態ではないことをいい、各瞬時においてはｒｕｎ状態やｒｅａｄｙ状態等と変化し得る。

【００３８】

ＡＰスレッド３５５ｂは、参照処理がなされているところの状態でありスレッドは停止している。参照処理は、ＡＰスレッドを強制的に停止状態にして、スタ

ック及びオブジェクト参照情報を参照して行われる。

ＡＰスレッド３５５ｃは、参照処理が未だなされていない状態であり、実行中である。

【００３９】

図７は、スレッド選定条件の内容を示す図である。

スレッド選定条件４２１は、どのスレッドを優先して参照処理の対象とするかの判断基準となる情報であり、スレッド状態４２２、スレッド優先度４２３及びスタックサイズ４２４から構成される。

スレッド状態４２２は、どの状態のスレッドを優先するかを示す情報であり、図７の例はｗａｉｔ状態のスレッドを、ｒｕｎ状態等のスレッドより優先する旨を示している。

【００４０】

スレッド優先度４２３は、スレッドについて定められている優先度の高いものを参照処理の対象として優先的に選定するか、又は低いものを優先的に選定するかを示す情報であり、図７の例では優先度の低いものを優先的に選定する旨を示している。

また、スタックサイズ４２４は、スレッドに対応するスタック領域の有効範囲の大きさが、大きいところのスレッドを優先的に選定するか、又は小さいところのスレッドを優先的に選定するかを示す情報であり、図７の例ではスタック領域の有効範囲の大きさが小さいものを優先的に選定する旨を示している。

【００４１】

＜動作＞

以下、ＧＣシステム１０の動作について説明する。

図８は、ＧＣ制御処理を示すフローチャートである。

ＧＣ制御処理は、タイマーに基づき一定周期で行われる。

ＧＣ制御部４１０は、最初に、ｆｒｏｍテーブルポインタ２１１とｔｏテーブルポインタ２１２の内容を交換することにより、ｆｒｏｍテーブルとｔｏテーブルとの内容を切り替える（ステップＳ１１）。これにより、現在解放されていない全てのオブジェクトを指す全てのオブジェクトポインタがｆｒｏｍテーブルに

格納されている状態になる。

【００４２】

続いて、ＧＣ制御部４１０は、スレッド制御部３１０にＡＰの全スレッドを停止させ（ステップＳ１２）、ＡＰの全スレッドについてのスレッド情報中のＧＣフラグ３２５をＯＮにし、現在のスタックポインタをスレッド情報中のＧＣ開始時スタックポインタ３２４に設定し（ステップＳ１３）、スレッド制御部３１０にＡＰの全スレッドの停止を解除させる（ステップＳ１４）。ステップＳ１３においては、更に、ＡＰのスレッド毎に、オブジェクト参照情報のメモリ領域を確保し、そのメモリ領域の先頭を指すポインタをオブジェクト参照情報先頭ポインタ３２６及びオブジェクト参照情報カレントポインタ３２７に設定する。

【００４３】

なお、スレッド制御部３１０は、従来のＯＳのマルチスレッド制御機構と同様の方式によりスレッドの停止及び停止解除を行う。このスレッドの停止は、スレッドを停止状態、即ちｓｌｅｅｐ状態にする制御であり、スレッドの停止解除とは、ｓｌｅｅｐ状態を解除して、そのｓｌｅｅｐの前の状態に戻す制御である。

ステップＳ１４に続いて、ＧＣ制御部４１０は、共有オブジェクト管理情報３５３内のオブジェクトポインタを辿ることで到達するオブジェクトに対するマーク付与を行うための共有オブジェクト参照処理を行い（ステップＳ１５）、ＧＣ未処理スレッドつまり未だ参照処理の対象として選定されていないスレッドが存在するか否かを判定する（ステップＳ１６）。なお、共有オブジェクト参照処理については後述する。

【００４４】

ステップＳ１６においてＧＣ未処理スレッドが存在すると判定した場合には、ＧＣ制御部４１０は、参照処理の対象となるスレッドを決定するための対象スレッド決定処理を行い（ステップＳ１７）、スレッド制御部３１０にその決定したスレッドを停止させ（ステップＳ１８）、その決定したスレッドに関しての参照処理を内容とする対象スレッド参照処理を実行し（ステップＳ１９）、その決定したスレッドのＧＣフラグ３２５をＯＦＦにし（ステップＳ２０）、スレッド制御部３１０にその決定したスレッドの停止を解除させて（ステップＳ２１）、再

度、ステップＳ１６の判定に戻る。なお、対象スレッド決定処理及び対象スレッド参照処理については、後述する。また、ステップＳ２０の直後に、ＧＣ制御部４１０は、対象スレッドに対応するオブジェクト参照情報のメモリ領域を解放する。

【００４５】

また、ステップＳ１６において、ＧＣ未処理スレッドが存在しないと判定した場合には、ＧＣ制御部４１０は、解放部４４０にマーク付与がなされていないオブジェクトを解放させ（ステップＳ２２）、ＧＣ制御処理を終える。ステップＳ２２においては、解放部４４０は、参照処理によってｔｏテーブルにオブジェクトポインタが移行されていないところのオブジェクト、つまりｆｒｏｍテーブルに残存しているオブジェクトポインタが指すオブジェクトに対応するメモリ領域を解放する。この解放は、オブジェクト生成部２３０が、オブジェクト生成のためにオブジェクトに割り当てるためのメモリ領域を、一般的なＯＳのメモリ管理機能に基づき確保することに対応し、その確保されたメモリ領域の解放を意味する。

【００４６】

図９は、対象スレッド決定処理を示すフローチャートである。

ＧＣ制御部４１０は、スレッド選定条件記憶部４２０内のスレッド選定条件４２１を参照して、対象スレッド決定処理を行う。

まず、ＧＣ制御部４１０は、ＡＰの各スレッドに対応するスレッド情報を参照して、状態３２１がｗａｉｔ状態のスレッドを検索する（ステップＳ３１）。この検索結果のスレッド数を判定し（ステップＳ３２）、１であれば、その検索結果のスレッドを参照処理の対象スレッドとして決定し（ステップＳ３７）、対象スレッド決定処理を終える。

【００４７】

また、ステップＳ３２において検索結果のスレッド数が０であれば、ＧＣ制御部４１０は、スレッドの優先度３２２が最も低いスレッドを検索し（ステップＳ３３）、検索結果のスレッド数は１か否かを判定する（ステップＳ３５）。また、ステップＳ３２において検索結果のスレッド数が２以上であれば、検索結果の

スレッドの中から更に絞り込むため、スレッドの優先度３２２が最も低いスレッドを検索し（ステップＳ３４）、検索結果のスレッド数は１か否かを判定する（ステップＳ３５）。

【００４８】

ステップＳ３５においてスレッド数が１と判定した場合には、ＧＣ制御部４１０は、その検索結果のスレッドを参照処理の対象スレッドとして決定し（ステップＳ３７）、対象スレッド決定処理を終える。

また、ステップＳ３５において、検索結果のスレッド数が１でないと判定した場合、即ち、最も低い優先度のスレッドが複数存在した場合には、ＧＣ制御部４１０は、検索結果のスレッドの中からスタックサイズが最小のスレッド検索し（ステップＳ３６）、スタックサイズが最小のスレッドを参照処理の対象スレッドとして決定し（ステップＳ３７）、対象スレッド決定処理を終える。

【００４９】

図１０は、共有オブジェクト参照処理を示すフローチャートである。

ＧＣ制御部４１０は、オブジェクトポインタ群を含む共有オブジェクト管理情報３５３の先頭のオブジェクトポインタに着目して（ステップＳ４１）、参照処理部４３０に参照処理（ステップＳ４２）を行わせることにより、ＡＰの実行環境としての仮想マシンにおいて必要なため確保されているオブジェクト群全てにマーク付与を行う。

【００５０】

図１１は、対象スレッド参照処理を示すフローチャートである。

ＧＣ制御部４１０は、対象スレッドに対応するスレッド情報中のＧＣ開始時スタックポインタ３２４が指す位置に着目し（ステップＳ５１）、参照処理部４３０に参照処理（ステップＳ５２）を行わせることにより、スタック領域内のオブジェクトポインタから辿ることのできる全てのオブジェクトへのマーク付与を行う。なお、ＧＣ制御処理（図８参照）の開始時点と、この対象スレッド参照処理を開始する時点は異なるため、ＧＣ開始時スタックポインタ３２４が指す位置とそれに続く各位置の内容は、ＧＣ開始時点とは異なっているが、後続するステップＳ５３及びステップＳ５４によって、ＧＣ開始以後に、対象スレッドの動作に

よって変化したオブジェクトポインタから辿ることのできる全てのオブジェクトへのマーク付与が行われるので、過剰にマーク付与をする場合はあり得るが、参照されているにも関わらずマーク付与が漏れるという事態は生じない。

【００５１】

続いて、ＧＣ制御部４１０は、オブジェクト参照情報先頭ポインタ３２６の指す位置に着目し（ステップＳ５３）、参照処理部４３０に参照処理（ステップＳ５４）を行わせ、対象スレッド参照処理を終える。

図１２は、参照処理を示すフローチャートである。

参照処理部４３０は、着目されているメモリ位置から、オブジェクトポインタを検索し（ステップＳ６１）、オブジェクトポインタを検出したか否かを判定し（ステップＳ６２）、検出した場合にはその検出したオブジェクトポインタと同値のオブジェクトポインタがｆｒｏｍテーブルに存在するか否かを判定し（ステップＳ６４）、ｆｒｏｍテーブルに存在すれば、ｆｒｏｍテーブルからｔｏテーブルにそのオブジェクトポインタをコピーして、ｆｒｏｍテーブル内のそのオブジェクトポインタが存在していた位置にｎｕｌｌ値を記録し（ステップＳ６６）、そのオブジェクトポインタが指すオブジェクトのデータ先頭に着目し（ステップＳ６７）、再度ステップＳ６１に戻る。

【００５２】

また、ステップＳ６４において、ｆｒｏｍテーブルにはそのオブジェクトポインタが存在しないと判定した場合には、既にｆｒｏｍテーブルからｔｏテーブルに移されているため、参照処理部４３０は、着目している位置の次の位置に着目し（ステップＳ６５）、再度ステップＳ６１に戻る。ステップＳ６５及びステップＳ６１の組み合わせにより、共有オブジェクト管理情報３５３内のオブジェクトポインタを次々と検索することや、スタック内のオブジェクトポインタを次々と検索することや、オブジェクト参照情報内のオブジェクトポインタを次々と検索することや、あるオブジェクト内のデータメンバとしてのオブジェクトポインタを次々と検索すること等が実現される。

【００５３】

また、ステップＳ６２においてオブジェクトポインタを検出できなかったと判

定した場合には、参照処理部４３０は、着目位置がオブジェクト内であるか否かを判定する（ステップＳ６３）。なお、オブジェクトポインタを検出できなかったというのは、着目位置が共有オブジェクト管理情報３５３内であればその共有オブジェクト管理情報３５３内においてオブジェクトポインタを検出できなかったということであり、着目位置がスタック内であればスタック内においてオブジェクトポインタを検出できなかったということであり、着目位置がオブジェクト参照情報内であればそのオブジェクト参照情報内においてオブジェクトポインタを検出できなかったということであり、着目位置がオブジェクト内であればそのオブジェクト内においてオブジェクトポインタを検出できなかったということである。

【００５４】

ステップＳ６３において着目位置がオブジェクト内であると判定した場合には、参照処理部４３０は、そのオブジェクト内に着目前の着目位置の次位置に着目し（ステップＳ６８）、再度ステップＳ６１に戻る。このステップＳ６８によって、参照処理部４３０は、ステップＳ６７によってオブジェクト内に着目する前に、着目していたオブジェクトポインタの次の位置に着目することになる。

【００５５】

また、ステップＳ６３において着目位置がオブジェクト内ではないと判定した場合には、参照処理部４３０は、参照処理を終える。

図１３は、命令実行処理を示すフローチャートである。

インタプリタ部１００における命令実行部１１０は、ｒｕｎ状態にされたＡＰのスレッドにおけるプログラム中の命令記述を逐次解釈して実行する命令実行処理を行う。

【００５６】

まず、命令実行部１１０は、スレッドの現在の実行位置における命令を解釈して実行し（ステップＳ７１）、オブジェクト参照検出部１２０は、そのスレッドに対応するスレッド情報中のＧＣフラグ３２５がＯＮであるか否かを判定し（ステップＳ７２）、ＯＮでなければ、命令実行部１１０が次の命令の解釈実行を行う（ステップＳ７１）。

【００５７】

ステップＳ７２において、ＧＣフラグ３２５がＯＮであると判定した場合には、オブジェクト参照検出部１２０は、ステップＳ７１において命令実行部１１０により実行された命令が、オブジェクトポインタを処理対象とする命令であったか否かを判定し（ステップＳ７３）、オブジェクトポインタを処理対象とする命令でなかったならば、命令実行部１１０が次の命令の解釈実行を行う（ステップＳ７１）。なお、オブジェクトポインタを処理対象とする命令とは、スタック内のオブジェクトポインタをオブジェクト内にコピーする演算や、オブジェクト内のオブジェクトポインタを他のオブジェクト内にコピーする演算である。

【００５８】

ステップＳ７３において、オブジェクトポインタを処理対象とする命令であったと判定した場合には、オブジェクト参照検出部１２０はそのオブジェクトポインタと同値のものがオブジェクト参照情報領域に既に格納済みであるか否かを判定し（ステップＳ７４）、格納済みである場合には、命令実行部１１０が次の命令の解釈実行を行う（ステップＳ７１）。

【００５９】

ステップＳ７４において、そのオブジェクトポインタと同値のものがオブジェクト参照情報のメモリ領域に格納済みでない場合には、オブジェクト参照検出部１２０は、そのオブジェクトポインタを、オブジェクト参照情報カレントポインタ３２７が指すオブジェクト参照情報中の位置に格納し（ステップＳ７５）、そのオブジェクトポインタの指すオブジェクトに着目して、オブジェクトチェーン追跡処理（ステップＳ７６）を行い、その後は命令実行部１１０が次の命令の解釈実行を行う（ステップＳ７１）。

【００６０】

なお、ステップＳ７１における実行対象とされる命令がオブジェクトを生成する命令である場合には、命令実行部１１０は、オブジェクト生成部２３０にオブジェクトの生成を指示し、これを受けてオブジェクト生成部２３０は、オブジェクトを生成するとともにそのオブジェクトを指すオブジェクトポインタをスレッドに対して返却し、そのオブジェクトポインタのコピーをｔｏテーブル内に設定

する。

【００６１】

図１４は、オブジェクトチェーン追跡処理を示すフローチャートである。

オブジェクト参照検出部１２０は、着目しているオブジェクトに未着目のオブジェクトポインタが存在するか否かを判定し（ステップＳ８１）、存在する場合には、その未着目の１つのオブジェクトポインタに着目し（ステップＳ８２）、その着目したオブジェクトポインタをオブジェクト参照情報のメモリ領域に格納し（ステップＳ８３）、その着目したオブジェクトポインタの指すオブジェクトに着目して更にオブジェクトチェーン処理（ステップＳ８１～Ｓ８４）を行う（ステップＳ８４）。なお、オブジェクトポインタをオブジェクト参照情報のメモリ領域に格納したときには、オブジェクト参照検出部１２０は、オブジェクト参照情報カレントポインタをそのオブジェクトポインタのサイズ分だけ進める。

【００６２】

また、ステップＳ８１において、着目しているオブジェクトに未着目のオブジェクトポインタが存在していないと判定した場合には、オブジェクト参照検出部１２０は、オブジェクトチェーン追跡処理を終える。

従って、この命令実行処理及びオブジェクトチェーン追跡処理によって、ＧＣフラグ３２５がＯＮである場合において、オブジェクトポインタが処理対象とされたときにはそのオブジェクトポインタから辿ることのできるオブジェクトポインタはいずれもオブジェクト参照情報のメモリ領域に格納されることになる。

【００６３】

なお、スレッドによるオブジェクトポインタを処理対象とする演算の実行によって、そのオブジェクトポインタで指されていたオブジェクトにそのスレッドからアクセスすることが不可能な状態となった場合、つまり、スレッドの動作により、（ａ）オブジェクトポインタがクリアされた場合、（ｂ）オブジェクトポインタが他の内容に更新された場合、或いは、（ｃ）オブジェクトポインタがスタック領域の有効範囲内に存在する場合においてスタックポインタの変更がなされてスタック領域の有効範囲外となりスレッドから基本的にアクセス不能となった場合にも、そのスレッドによって予めそのオブジェクトポインタが他のスレッド

からはアクセス可能な場所にコピーされていたならば、そのオブジェクトポインタで指されるオブジェクトにはマーク付与が必要となるので、そのために、上述したステップＳ７２～Ｓ７６、ステップＳ８１～Ｓ８４の処理が行われる。

＜実施形態２＞

以下、本発明の実施形態２に係るＧＣシステムについて図面を用いて説明する。

【００６４】

実施形態２に係るＧＣシステムは、ＡＰのスレッドを順次停止して参照処理を行う点で、実施形態１に係るＧＣシステム１０と基本的に同様である。ただし、実施形態２に係るＧＣシステムは、主に、スレッド毎にオブジェクト参照情報やＧＣフラグを有するのではなくシステム全体でまとめてオブジェクト参照情報やＧＣフラグを有する点や、スレッド情報の内容を一般的なマルチスレッド制御に要する程度に抑えている点が、ＧＣシステム１０と異なる。

【００６５】

図１５は、本発明の実施形態２に係るＧＣシステムの機能ブロック図である。

図１５に示すＧＣシステム２０の構成要素のうち、実施形態１で示したＧＣシステム１０と同様の構成要素については、図１と同じ符号を付しており、ここではＧＣシステム２０に特有の点を中心に説明する。なお、特に説明しない点については、ＧＣシステム２０はＧＣシステム１０と同様である。

【００６６】

ＧＣシステム２０は、図１５に示すように、インタプリタ部１１００、オブジェクト管理部２００、スレッド管理部１３００及びＧＣ部１４００を備える。

ここで、インタプリタ部１１００は、基本的にインタプリタであってＡＰを実行する機能を担い、命令実行部１１０及びオブジェクト参照検出部１１２０を有する。

【００６７】

スレッド管理部１３００は、マルチスレッド制御を行いスレッドを管理する機能を担い、スレッド制御部３１０と、スレッド毎に、スレッド情報１３２０及びスタック３４０に対応するメモリ領域を有する。なお、スレッド情報１３２０は

、図１６に示すように状態３２１、優先度３２２及びスタックポインタ３２３を含むものである。

【００６８】

ＧＣ部１４００は、基本的にいわゆるガーベジコレクタに相当し、ＧＣ制御部１４１０と、スレッド選定条件記憶部４２０と、参照処理部１４３０と、解放部４４０と、オブジェクト参照情報１４５０及びＧＣフラグ１４６０に対応するメモリ領域とを有する。

インタプリタ部１１００におけるオブジェクト参照検出部１２０は、ＧＣフラグ１４６０がＯＮである場合に限って、命令実行部１１０により実行されるスレッド内の命令が、オブジェクトのオブジェクトポインタを処理対象とする命令であるときに、そのオブジェクトポインタを、オブジェクト参照情報１４５０のメモリ領域の中に格納する機能を有する。

【００６９】

ＧＣ部１４００を中心として行われるＧＣは、基本的にマークアンドスイープ方式を用い、参照処理の対象となるＡＰスレッドを順番に選択して、選択したスレッドを停止させて参照処理を行ってからその停止を解除する方式のものである。

ＧＣ部１４００における参照処理部１４３０は、スタック及びオブジェクト参照情報１４５０を参照することにより参照処理を行う機能を有する。オブジェクト参照情報１４５０は、ＧＣシステム１０におけるオブジェクト参照情報と同様にオブジェクトポインタを内容とする情報であるが、スレッド毎に存在するのではない。このオブジェクト参照情報１４５０のメモリ領域には、ＡＰの全スレッドについてのオブジェクト参照検出部１１２０により、オブジェクトポインタが命令の処理対象とされた場合にそのオブジェクトポインタが格納される。

【００７０】

また、ＧＣ制御部１４１０は、図１７に示すＧＣ制御処理を実行する機能を有する。

図１７は、実施形態２におけるＧＣ制御処理を示すフローチャートである。なお、このフローチャート中の処理ステップのうち、実施形態１のＧＣ制御処理と

同様のものについては、図８と同じ符号を付して示している。

【００７１】

このＧＣ制御処理は、タイマーに基づき一定周期で行われる。

ＧＣ制御部１４１０は、最初に、ｆｒｏｍテーブルポインタ２１１とｔｏテーブルポインタ２１２の内容を交換することにより、ｆｒｏｍテーブルとｔｏテーブルとの内容を切り替え（ステップＳ１１）、ＧＣフラグ１４６０をＯＮにする（ステップＳ１１１）。

【００７２】

ステップＳ１１１に続いて、ＧＣ制御部１４１０は、共有オブジェクト管理情報３５３内のオブジェクトポインタを辿ることで到達するオブジェクトに対するマーク付与を行うための共有オブジェクト参照処理を行い（ステップＳ１５）、ＧＣ未処理スレッドつまり未だ参照処理の対象として選定されていないスレッドが存在するか否かを判定する（ステップＳ１６）。

【００７３】

ステップＳ１６においてＧＣ未処理スレッドが存在すると判定した場合には、ＧＣ制御部１４１０は、参照処理の対象となるスレッドを決定するための対象スレッド決定処理を行い（ステップＳ１７）、スレッド制御部３１０にその決定したスレッドを停止させ（ステップＳ１８）、その決定したスレッドに対応するスタックポインタの位置に着目して（ステップＳ１１２）、参照処理部１４３０に参照処理（図１２参照）を実行させ（ステップＳ１１３）、スレッド制御部３１０にその決定したスレッドの停止を解除させて（ステップＳ２１）、再度、ステップＳ１６の判定に戻る。

【００７４】

また、ステップＳ１６において、ＧＣ未処理スレッドが存在しないと判定した場合には、ＧＣ制御部１４１０は、スレッド制御部３１０にＡＰの全スレッドを停止させ（ステップＳ１１４）、オブジェクト参照情報１４５０のメモリ領域の先頭位置に着目し（ステップＳ１１５）、参照処理部１４３０に参照処理を実行させ（ステップＳ１１６）、ＧＣフラグ１４６０をＯＦＦにし（ステップＳ１１７）、スレッド制御部３１０にＡＰの全スレッドの停止を解除させ（ステップＳ

１１８）、解放部４４０にマーク付与がなされていないオブジェクトを解放させ（ステップＳ２２）、ＧＣ制御処理を終える。

【００７５】

ここで、インタプリタ部１１００によりなされる命令実行処理について簡単に説明する。

図１８は、実施形態２における命令実行処理を示すフローチャートである。

同図に示すように、この命令実行処理は、図１３で示した命令実行処理からステップＳ７６を削除したものと基本的に等しい。

【００７６】

まず、命令実行部１１０は、スレッドの現在の実行位置における命令を解釈して実行し（ステップＳ７１）、オブジェクト参照検出部１１２０は、ＧＣフラグ１４６０がＯＮであるか否かを判定し（ステップＳ７２）、ＯＮでなければ、命令実行部１１０が次の命令の解釈実行を行う（ステップＳ７１）。

ステップＳ７２において、ＧＣフラグ３２５がＯＮであると判定した場合には、オブジェクト参照検出部１１２０は、ステップＳ７１において命令実行部１１０により実行された命令が、オブジェクトポインタを処理対象とする命令であったか否かを判定し（ステップＳ７３）、オブジェクトポインタを処理対象とする命令でなかったならば、命令実行部１１０が次の命令の解釈実行を行う（ステップＳ７１）。

【００７７】

ステップＳ７３において、オブジェクトポインタを処理対象とする命令であったと判定した場合には、オブジェクト参照検出部１１２０はそのオブジェクトポインタと同値のものがオブジェクト参照情報領域に既に格納済みであるか否かを判定し（ステップＳ７４）、格納済みである場合には、命令実行部１１０が次の命令の解釈実行を行う（ステップＳ７１）。

【００７８】

ステップＳ７４において、そのオブジェクトポインタと同値のものがオブジェクト参照情報のメモリ領域に格納済みでない場合には、オブジェクト参照検出部１１２０は、そのオブジェクトポインタを、オブジェクト参照情報中の位置に格

納し（ステップＳ７５）、その後は命令実行部１１０が次の命令の解釈実行を行う（ステップＳ７１）。

【００７９】

このようにしてＧＣ制御部１４１０は、スレッド選定条件を参照することにより、参照処理の対象にするＡＰスレッドを１つ選定し、選択したスレッドについて、スレッド制御部３１０にそのスレッドを停止させてから、参照処理部１４３０にそのスレッドに対応するスタックに基づいて参照処理を行わせ、その後にスレッド制御部３１０に選択したスレッドの停止を解除させてから、次のＡＰスレッドの選択を行うという手順を、未処理のＡＰスレッドが存在しなくなるまで繰り返し、その後、スレッド制御部３１０にＡＰの全スレッドを停止させ、参照処理部１４３０にオブジェクト参照情報に基づいて参照処理を行わせて、スレッド制御部３１０にＡＰの全スレッドの停止を解除させて、解放部４４０に、マーク付与がなされていないオブジェクトの解放を行わせる。

＜補足＞

以上、本発明に係るＧＣシステムについて、実施形態１、２に基づいて説明したが、本発明は、勿論これらの実施形態に限られない。以下、実施形態の変形に関して説明する。

（１）スレッド選定条件は、スレッド状態、スレッド優先度、スタックサイズという３つの条件をこの順に優先適用されることとしたが、条件数や適用順序等は、これに限定されることはなく、また、例えばスレッド優先度が高いものを条件にしたりスタックサイズが大きいものを条件にしたりしてもよい。

【００８０】

但し、実施形態において示したようにスレッド状態がｗａｉｔ状態のものを参照処理対象と選定することは、現在動作しているＡＰに対する悪影響が少ないという効果につながる。また、参照処理（図１２）のアルゴリズムにより、あるスレッドに対する参照処理にかかる時間は、参照処理を先に行うスレッドより後に行うスレッドの方が比較的短くなるため、実施形態において示したようにスレッド優先度が低いものを早めに参照処理対象と選定するようにしておくことで、即応性の求められるところのスレッド優先度の高いスレッドの停止時間を短くし得

るという効果が生じる。また、同様の理由と、スタックサイズが大きいほど参照処理には時間がかかるという理由とから、対応するスタックサイズの小さいスレッドから順に参照処理を行うようにすることで、同じスレッド優先度のスレッド間である程度等しくなるようにスレッドの停止時間を分散できるという効果が生じる。

（２）実施形態では、ＧＣ制御処理がタイマーに基づき一定周期で行われることとしたが、これに限定されることはなく、ＡＰに割り当てられる空きメモリの量が所定量より少なくなった場合にＧＣ制御処理を行うようにしてもよい。

（３）実施形態で示したオブジェクト参照情報のメモリ領域は、連続した物理アドレスで表されるメモリ領域であってもよいし、オブジェクトポインタの格納に際して必要が生じた度に一定量のメモリ領域を追加的に確保してその確保した複数の断続的なメモリ領域を論理アドレス上で連続しているように扱うこととしてもよい。

（４）実施形態１で示したＧＣシステムにおいては、インタプリタによる命令実行処理中に、ＧＣフラグがＯＮであればオブジェクトポインタを処理対象としたときにオブジェクト参照情報のメモリ領域にそのオブジェクトポインタを保存する処理を行っておくこととし、命令実行処理中では、そのオブジェクトポインタと同値のものをｆｒｏｍテーブルからｔｏテーブルに移行することを内容とする参照処理を行わずインタプリタによる命令実行の高速性を保つようにしたが、命令実行処理中にその参照処理を行うこととしてもよい。

（５）実施形態では共有オブジェクト管理情報内のオブジェクトポインタから辿ることのできるオブジェクトへのポインタは参照処理によってｆｒｏｍテーブルからｔｏテーブルに移行されることとしたが、参照処理を行わなくても、共有オブジェクト管理情報によって管理されているオブジェクトは解放部による解放対象から除外しさえすればよい。

（６）実施形態で示したＧＣシステムは、コンピュータ上に実装されるのみならず、ＣＰＵを備える携帯端末、家電機器等において実装されることとしてもよい。

（７）実施形態で示したＧＣシステムの各機能を実現させるための各種処理（図

８～図１４、図１７、図１８参照）をＣＰＵに実行させるためのプログラムを、記録媒体に記録し又は各種通信路等を介して、流通させ頒布することもできる。このような記録媒体には、ＩＣカード、光ディスク、フレキシブルディスク、ＲＯＭ等がある。流通、頒布されたプログラムは、ＣＰＵを備える装置におけるＣＰＵで読み取り可能なメモリ等に格納されることにより利用に供され、そのＣＰＵがそのプログラムを実行することにより実施形態で示したＧＣシステムの各機能が実現される。

【００８１】

【発明の効果】

以上説明したように、本発明に係るＧＣシステムは、複数スレッドで構成されるオブジェクト指向プログラムの実行過程において不要になったオブジェクトに対応するメモリ領域を解放するガーベジコレクションシステムであって、複数スレッドそれぞれを順に選択する選択手段と、選択されたスレッドについて、まず当該スレッドの実行を停止して、その状態で、当該スレッドからオブジェクトポインタの参照を通じてアクセス可能であるオブジェクトを検出して当該検出したオブジェクトを非解放対象として管理して、その後に、当該スレッドの実行を再開するという一連の手順からなる検査処理を実施する検査手段と、前記選択手段による前記選択が開始された後において、実行中のスレッドによりオブジェクトポインタが処理対象にされたことを検知した場合には、当該オブジェクトポインタが指すオブジェクトを非解放対象として管理する検知手段と、前記複数スレッド全てについて前記検査処理が完了した後において、非解放対象として管理されているオブジェクト以外のオブジェクトに対応するメモリ領域を、解放する解放手段とを備えることを特徴とする。

【００８２】

ここで、オブジェクトを非解放対象として管理するとは、例えばそのオブジェクトポインタをｆｒｏｍテーブルからｔｏテーブルに移行する上述の方法等により、マーク付与を実現することをいう。

これにより、ＡＰスレッドからスタック内やオブジェクト内のオブジェクトポインタを介して辿ることができるオブジェクトについての非解放対象としての指

定の過程つまりマーク付与の過程においては、そのＡＰスレッドを停止させているため、そのＡＰスレッドが動作することに起因してスタック領域内のデータが変化するという事態が起こらないため、その過程におけるマーク付与が無駄になる危険性はなくなり、この点についてはＣＰＵの利用効率低下を防止できる。

【００８３】

また、これにより、全てのＡＰスレッドを停止して上述のマーク付与を行っているのではないため、ＡＰの全スレッドが停止している期間の長期化が防止できる。なお、全ＡＰスレッドを停止させないことによるオブジェクトへの参照状態の変化に対しては、実行中のスレッドによりオブジェクトポインタが処理対象とされたことを監視して対応しているため、いずれかのスレッドからアクセス可能なオブジェクトについては必ず非解放対象として管理されることになる。

【００８４】

また、前記検知手段は、前記検知を、実行中のスレッドについて未だ検査処理がなされていない場合に限り行い、前記検知手段は、前記検知をした場合に、当該実行中のスレッドに処理対象にされたオブジェクトポインタ、及び当該オブジェクトポインタから辿ることのできるオブジェクト内のオブジェクトポインタを、当該スレッドに対応する作業用メモリ領域に保存する検出部と、前記検査手段によりスレッドの実行が停止されている間に、当該スレッドに対応する作業用メモリ領域内のオブジェクトポインタから辿ることのできるオブジェクトを非解放対象として管理する管理部とを有することとしてもよい。

【００８５】

これにより、スレッドが、参照処理つまり参照されているオブジェクトへのマーク付与の処理の対象とされるまでの間だけ、インタプリタによるそのスレッドの実行時にオブジェクトポインタが処理対象とされたことを検出して作業用メモリ領域、即ちオブジェクト参照情報のメモリ領域に格納する処理を行うため、一旦スレッドが参照処理の対象とされた後は、そのスレッドはインタプリタによる実行時に、その検出がない分だけ、より高速に動作するようになる。

【００８６】

また、前記検査処理は、選択された当該スレッドに対応するスタック内のオブ

ジェクトポインタが指すオブジェクトを、前記アクセス可能であるとして検出した際において、検出した当該オブジェクトが既に非解放対象として管理されておらず、かつ、当該オブジェクト内にオブジェクトポインタがあるときに限り、当該オブジェクトポインタが指すオブジェクトを更に前記アクセス可能であるとして検出する手順を、繰り返し行うことを内容とする処理であり、前記選択手段は、最初の選択後は、前記検査手段により前記検査処理が行われた後において前記複数スレッドのうち前記検査処理を実施されていないスレッドがある限り更なる選択を行い、前記選択手段は、各スレッドに関する情報を参照し、所定のスレッド選定条件に基づき、前記選択を行うこととしてもよい。

【００８７】

これにより、既に非管理対象として管理されたオブジェクトの配下のオブジェクトを重複的に検出のための処理の対象とすることを防止することができる。また、この重複的に検出のための処理を行わない仕組みにより、参照処理の対象として早期に選択されたスレッドより、遅く選択されたスレッドの方が参照処理に係る時間が短縮される可能性が高まるので、各スレッドに要求される応答性能に鑑みて、予めスレッド選定条件を定めておくことで、各スレッドが適切な応答性能を発揮して実行されるようにある程度制御することができるようになる。

【００８８】

また、前記スレッド選定条件は、スレッド状態がｗａｉｔ状態であるスレッドをスレッド状態がｗａｉｔ状態以外であるスレッドよりも早期に選択することを示す条件を含み、前記選択手段は、前記選択を行う時点でｗａｉｔ状態のスレッドがある限り、当該ｗａｉｔ状態のスレッドを選択することとしてもよい。

これにより、ｗａｉｔ状態のスレッドを停止することになるため、現在動作しているＡＰスレッドに対する悪影響の発生が抑えられる。

【００８９】

また、前記スレッド選定条件は、スレッド優先度のスレッド優先度の低いスレッドをスレッド優先度の高いスレッドよりも早期に選択することを示す条件を含むこととしてもよい。

これにより、スレッド優先度の高いスレッドは、参照処理が短時間でなされる

可能性が高くなり、実行性能の低下がある程度防止されるようになる。

【００９０】

また、前記スレッド選定条件は、スレッドに対応するスタックサイズが小さいスレッドをスタックサイズが大きいスレッドよりも早期に選択することを示す条件を含むこととしてもよい。

これにより、スレッドからアクセス可能なオブジェクトへのオブジェクトポインタを格納するスタックの有効範囲が小さい程、参照処理に要する時間は短くなる傾向にあることに鑑みれば、各スレッドの停止時間をある程度均等化する効果が得られ、特定のスレッドの応答性が他と比べて際立って悪くなるという事態がある程度回避できるようになる。

【図面の簡単な説明】

【図１】

本発明の実施形態１に係るＧＣシステムの機能ブロック図である。

【図２】

オブジェクトとオブジェクトポインタとの関係を例示する図である。

【図３】

ｆｒｏｍテーブル及びｔｏテーブルを示す図である。

【図４】

参照処理によるｆｒｏｍテーブル及びｔｏテーブルの内容の変化を示す図である。

【図５】

スレッド情報、スタック及びオブジェクト参照情報を示す図である。

【図６】

参照処理とＡＰスレッドの状態との関係を示す図である。

【図７】

スレッド選定条件の内容を示す図である。

【図８】

ＧＣ制御処理を示すフローチャートである。

【図９】

対象スレッド決定処理を示すフローチャートである。

【図１０】

共有オブジェクト参照処理を示すフローチャートである。

【図１１】

対象スレッド参照処理を示すフローチャートである。

【図１２】

参照処理を示すフローチャートである。

【図１３】

命令実行処理を示すフローチャートである。

【図１４】

オブジェクトチェーン追跡処理を示すフローチャートである。

【図１５】

本発明の実施形態２に係るＧＣシステムの機能ブロック図である。

【図１６】

スレッド情報及びスタックを示す図である。

【図１７】

実施形態２におけるＧＣ制御処理を示すフローチャートである。

【図１８】

実施形態２における命令実行処理を示すフローチャートである。

【符号の説明】

１００、１１００　インタプリタ部

１１０　命令実行部

１２０、１１２０　オブジェクト参照検出部

２００　オブジェクト管理部

２１０　オブジェクト情報記憶部

２１１　ｆｒｏｍテーブルポインタ

２１２　ｔｏテーブルポインタ

２２１　ｆｒｏｍテーブル

２２２　ｔｏテーブル

２３０　オブジェクト生成部

２４０　テーブル切替部

３００、１３００　スレッド管理部

３１０　スレッド制御部

３２０、１３２０　スレッド情報

３３０　オブジェクト参照情報

３４０　スタック

４００、１４００　ＧＣ部

４１０、１４１０　ＧＣ制御部

４２０　スレッド選定条件記憶部

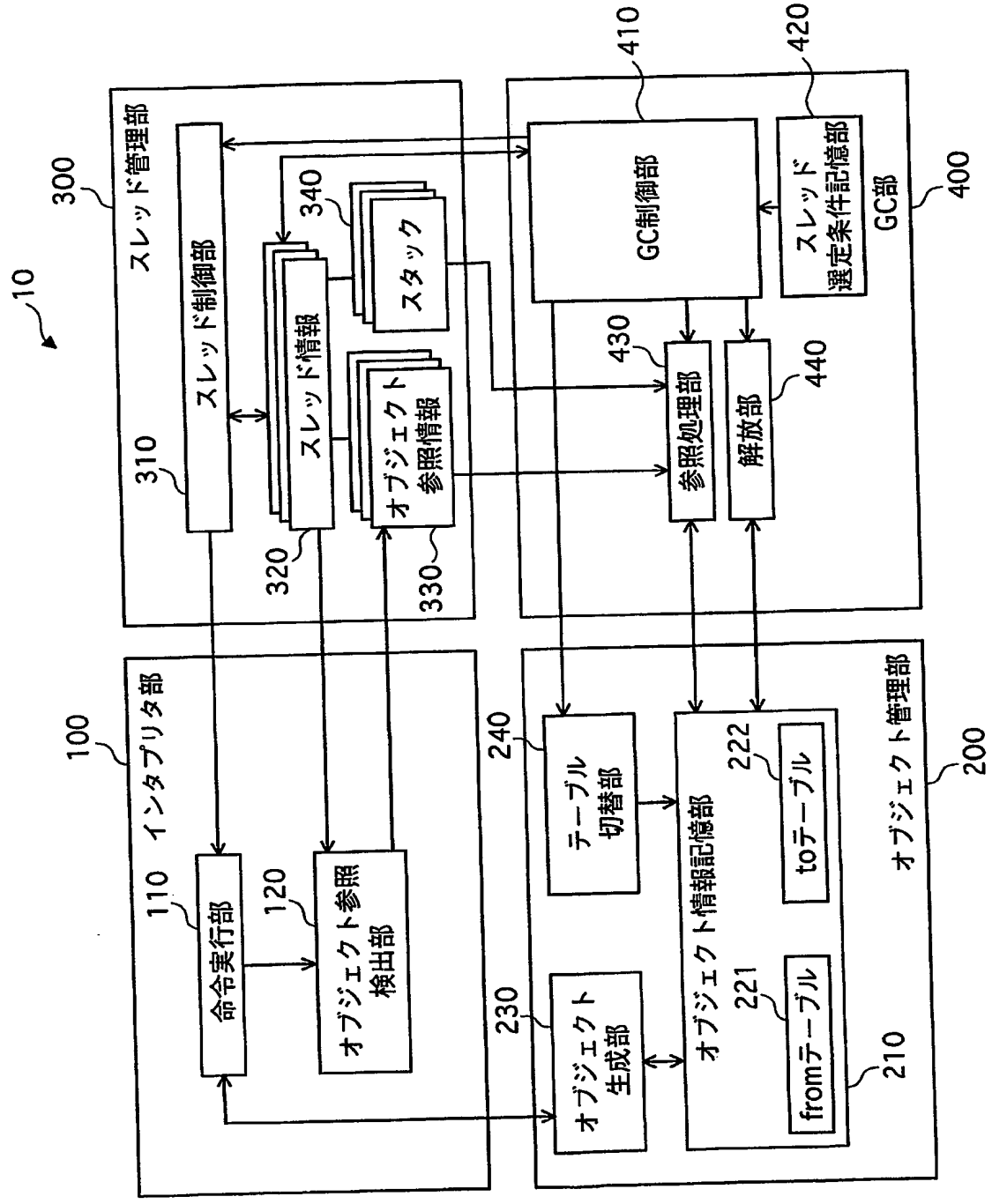
４３０、１４３０　参照処理部

４４０　解放部

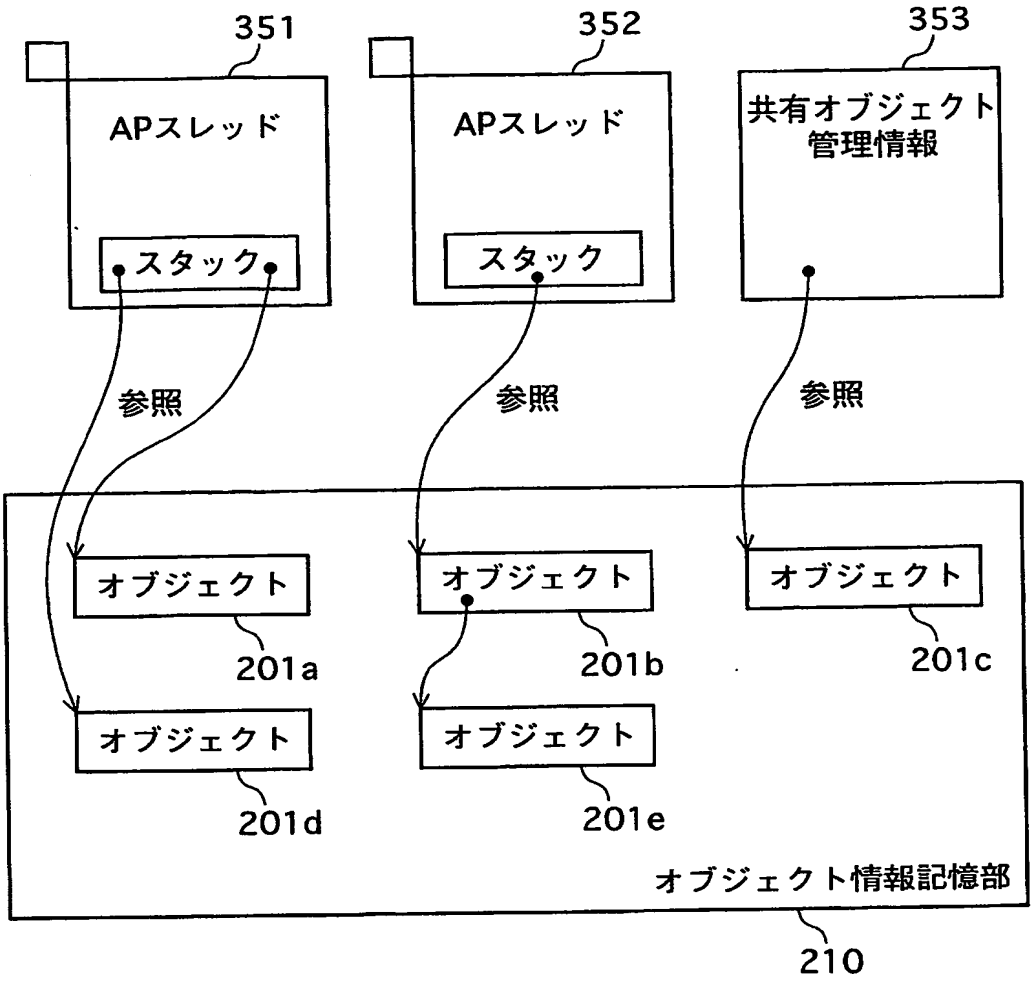
１４５０　オブジェクト参照情報

【書類名】　図面

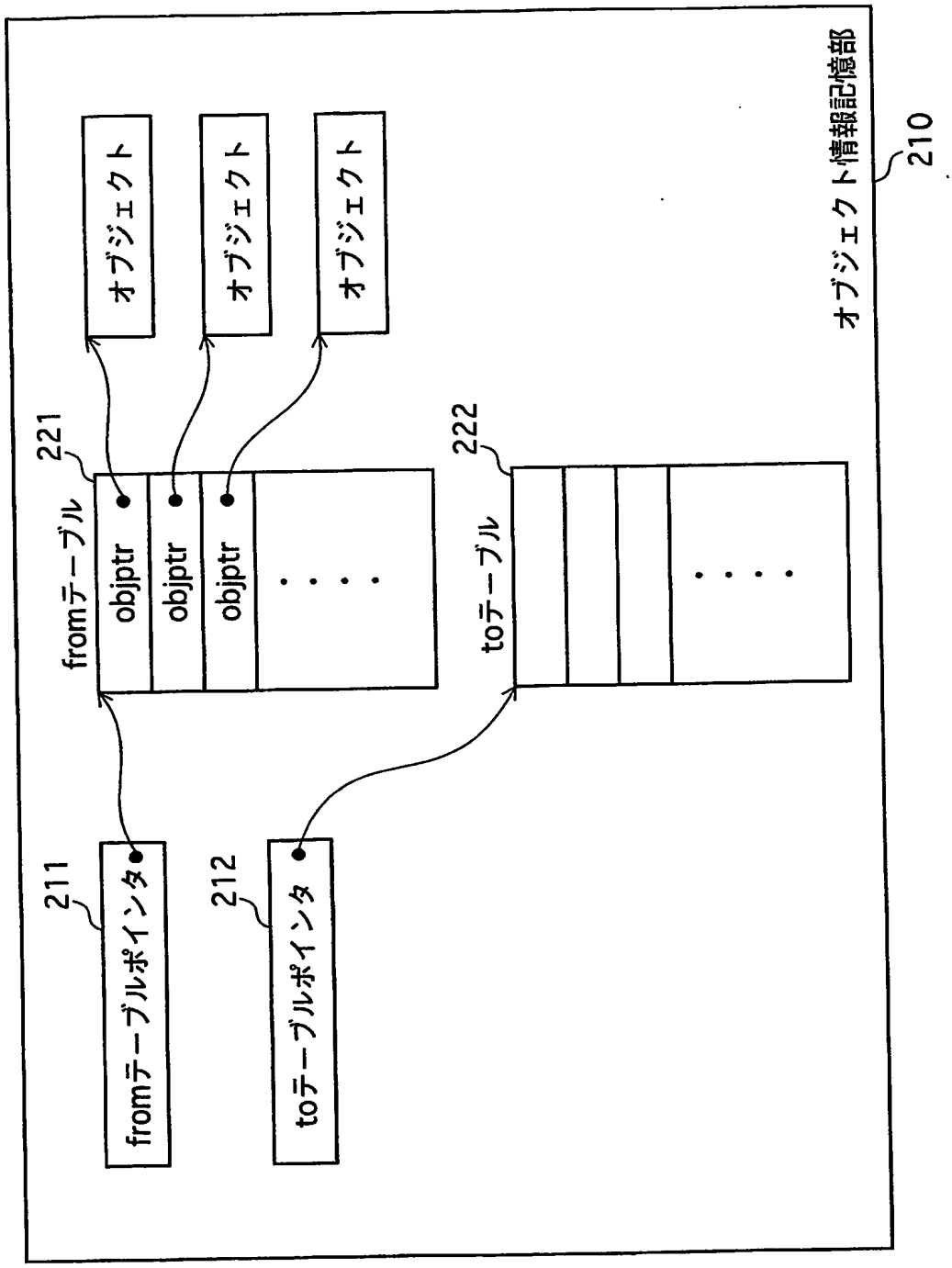
【図１】


10
300
スレッド管理部
スレッド制御部
310
スレッド情報
320
スタック
340
オブジェクト
参照情報
330
410
GC制御部
420
スレッド
選定条件記憶部
400
GC部
430
参照処理部
440
解放部
100
インタプリタ部
110
命令実行部
120
オブジェクト参照
検出部
240
テーブル
切替部
222
toテーブル
オブジェクト情報記憶部
221
fromテーブル
230
オブジェクト
生成部
210
200
オブジェクト管理部

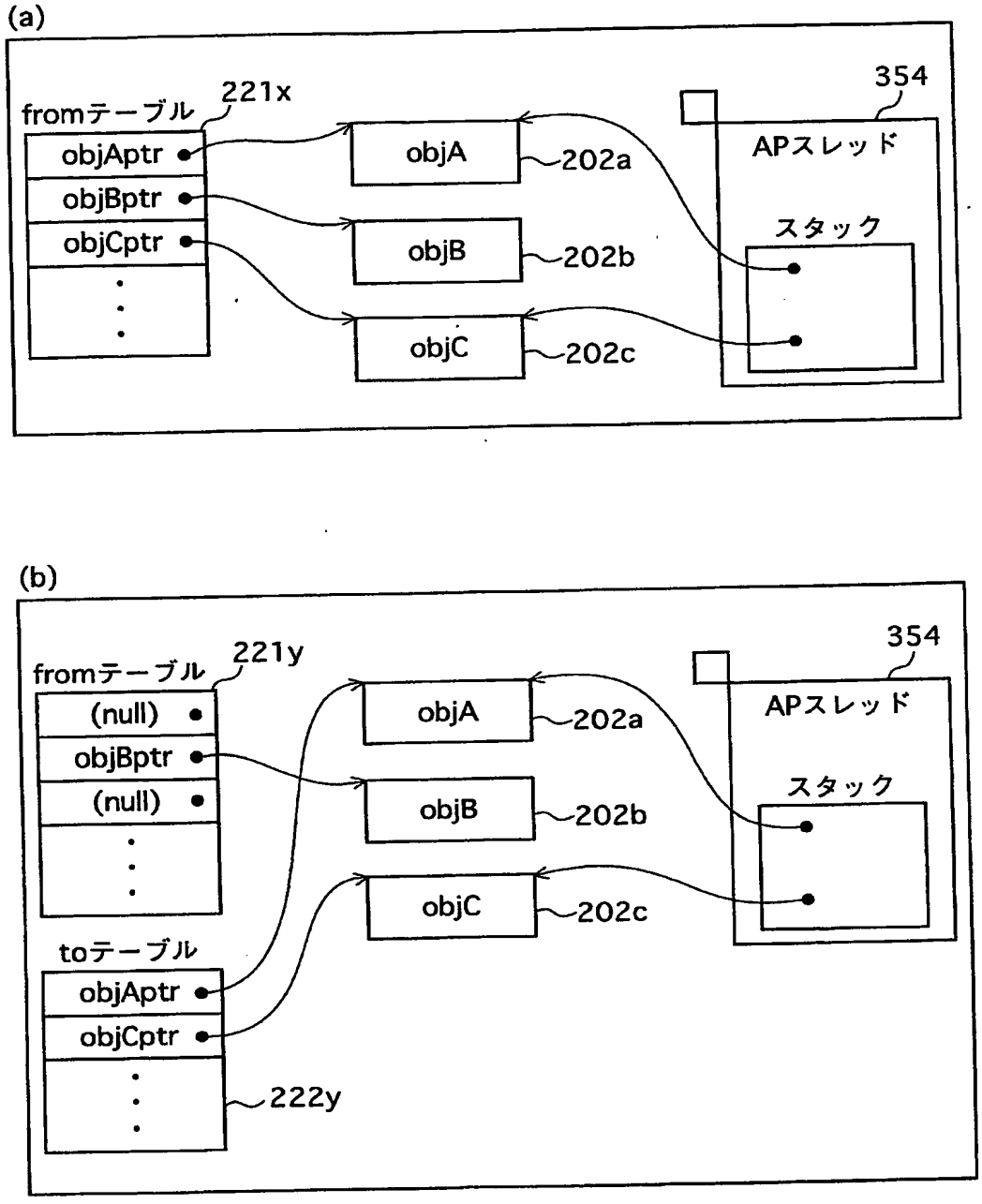
【図２】


351
352
353
APスレッド
APスレッド
共有オブジェクト
管理情報
スタック
スタック
参照
参照
参照
オブジェクト
オブジェクト
オブジェクト
201a
201b
201c
オブジェクト
オブジェクト
201d
201e
オブジェクト情報記憶部
210

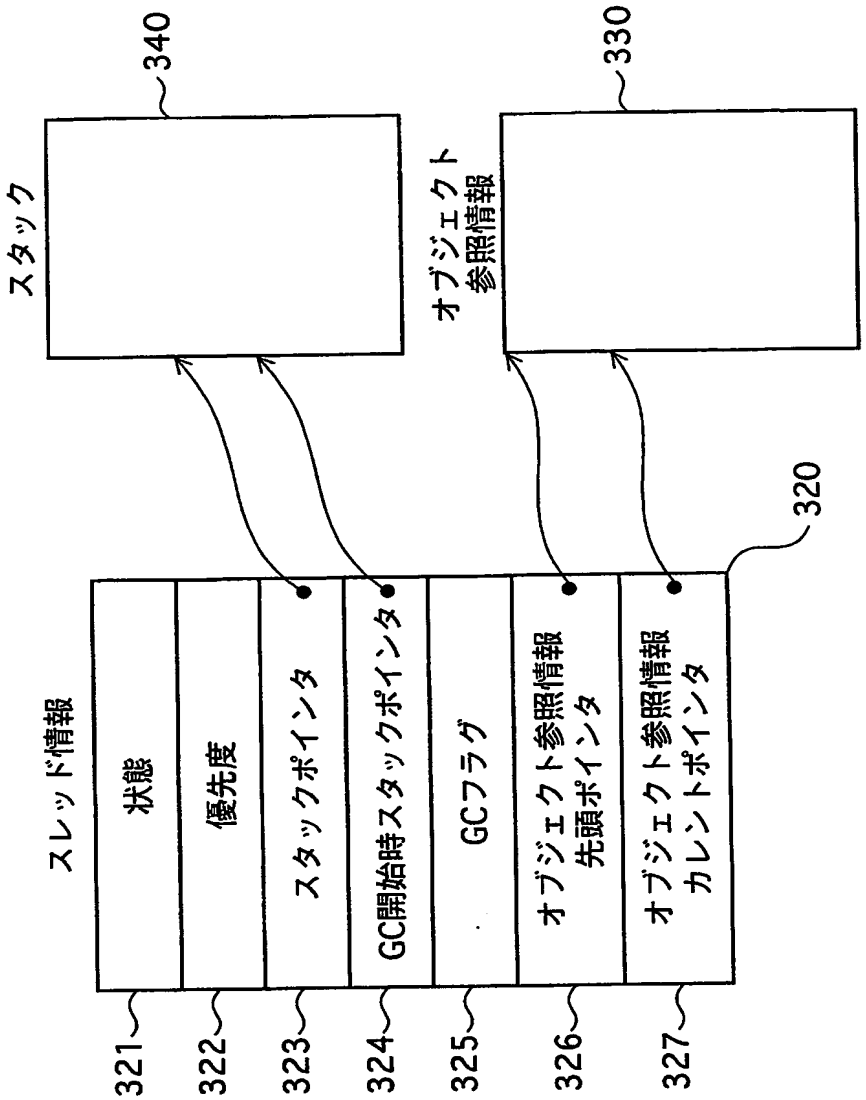
【図３】


オブジェクト情報記憶部
210
オブジェクト
オブジェクト
オブジェクト
221
fromテーブル
objptr
objptr
objptr
222
toテーブル
211
fromテーブルポインタ
212
toテーブルポインタ

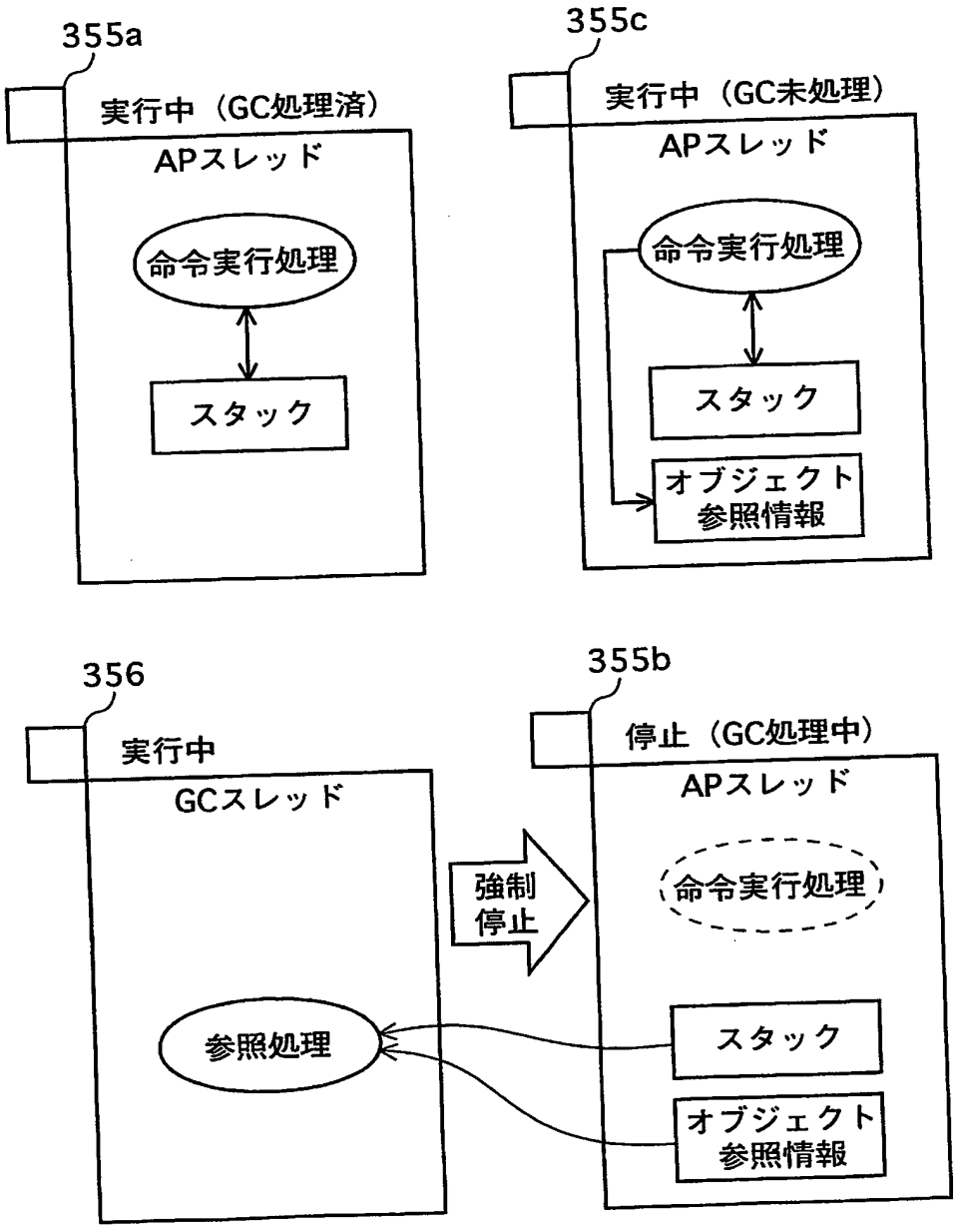
【図４】


(a)
fromテーブル
221x
objAptr
objBptr
objCptr
objA
202a
objB
202b
objC
202c
354
APスレッド
スタック
(b)
fromテーブル
221y
(null)
objBptr
(null)
objA
202a
objB
202b
objC
202c
354
APスレッド
スタック
toテーブル
objAptr
objCptr
222y

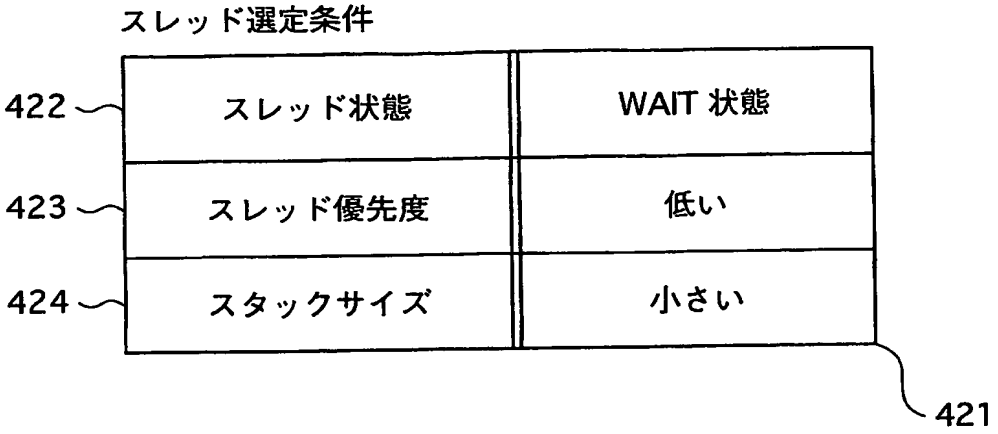
【図５】


340
スタック
330
オブジェクト
参照情報
320
スレッド情報
状態
優先度
スタックポインタ
GC開始時スタックポインタ
GCフラグ
オブジェクト参照情報
先頭ポインタ
オブジェクト参照情報
カレントポインタ
321
322
323
324
325
326
327

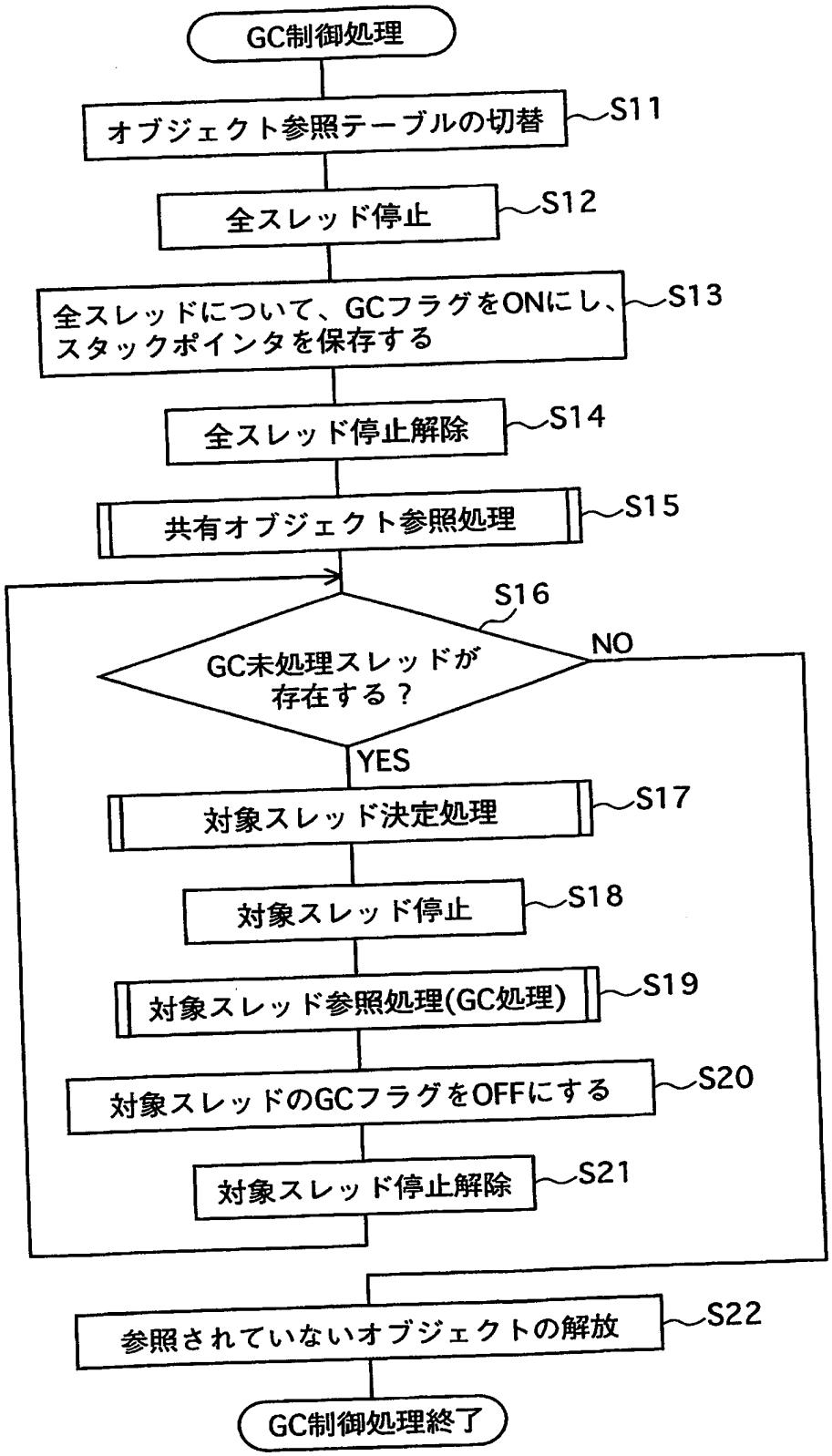
【図６】


355a
実行中（GC処理済）
APスレッド
命令実行処理
スタック
355c
実行中（GC未処理）
APスレッド
命令実行処理
スタック
オブジェクト参照情報
356
実行中
GCスレッド
参照処理
強制停止
355b
停止（GC処理中）
APスレッド
命令実行処理
スタック
オブジェクト参照情報

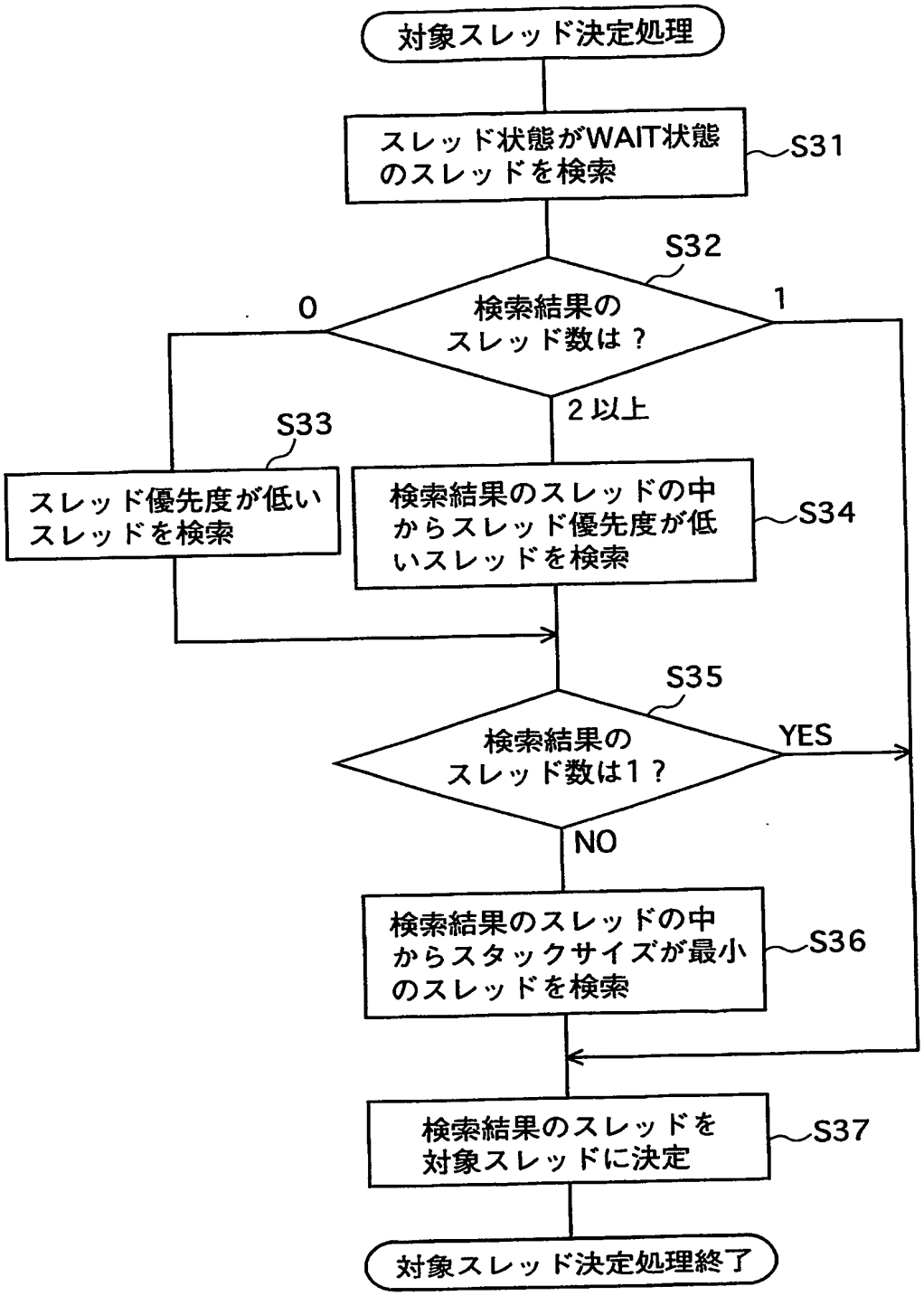
【図７】


スレッド選定条件
422
スレッド状態
WAIT 状態
423
スレッド優先度
低い
424
スタックサイズ
小さい
421

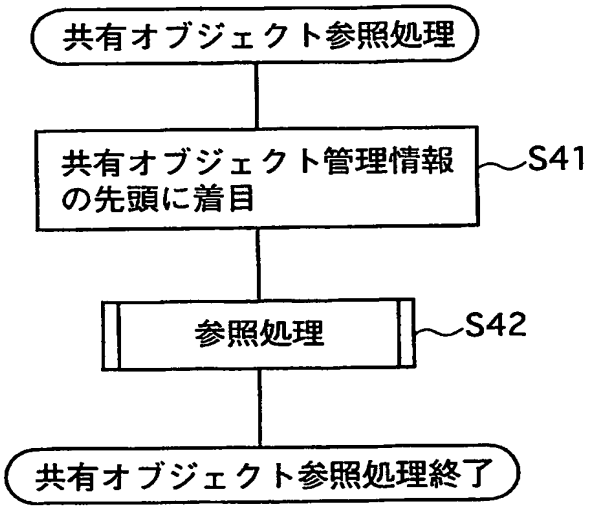
【図８】


GC制御処理
オブジェクト参照テーブルの切替
S11
全スレッド停止
S12
全スレッドについて、GCフラグをONにし、スタックポインタを保存する
S13
全スレッド停止解除
S14
共有オブジェクト参照処理
S15
S16
GC未処理スレッドが存在する？
NO
YES
対象スレッド決定処理
S17
対象スレッド停止
S18
対象スレッド参照処理(GC処理)
S19
対象スレッドのGCフラグをOFFにする
S20
対象スレッド停止解除
S21
参照されていないオブジェクトの解放
S22
GC制御処理終了

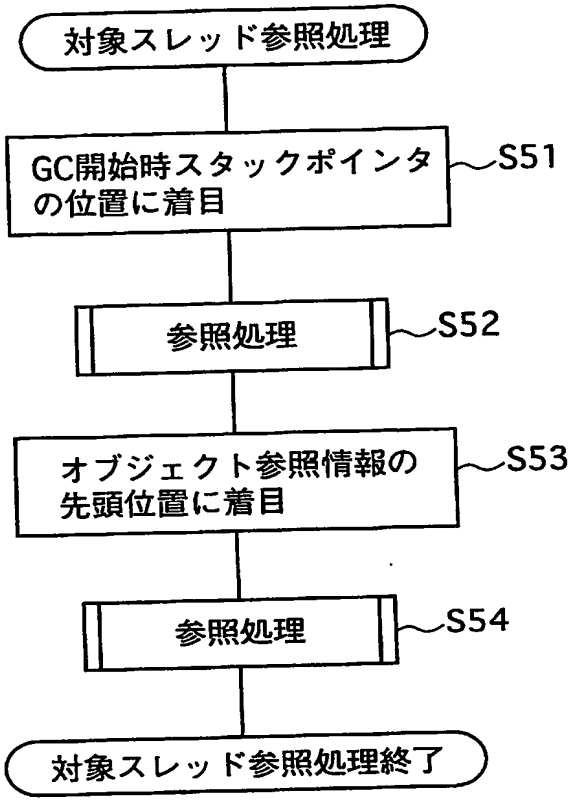
【図９】


対象スレッド決定処理
スレッド状態がWAIT状態のスレッドを検索
S31
S32
検索結果のスレッド数は？
0
1
2以上
S33
スレッド優先度が低いスレッドを検索
検索結果のスレッドの中からスレッド優先度が低いスレッドを検索
S34
S35
検索結果のスレッド数は1？
YES
NO
検索結果のスレッドの中からスタックサイズが最小のスレッドを検索
S36
検索結果のスレッドを対象スレッドに決定
S37
対象スレッド決定処理終了

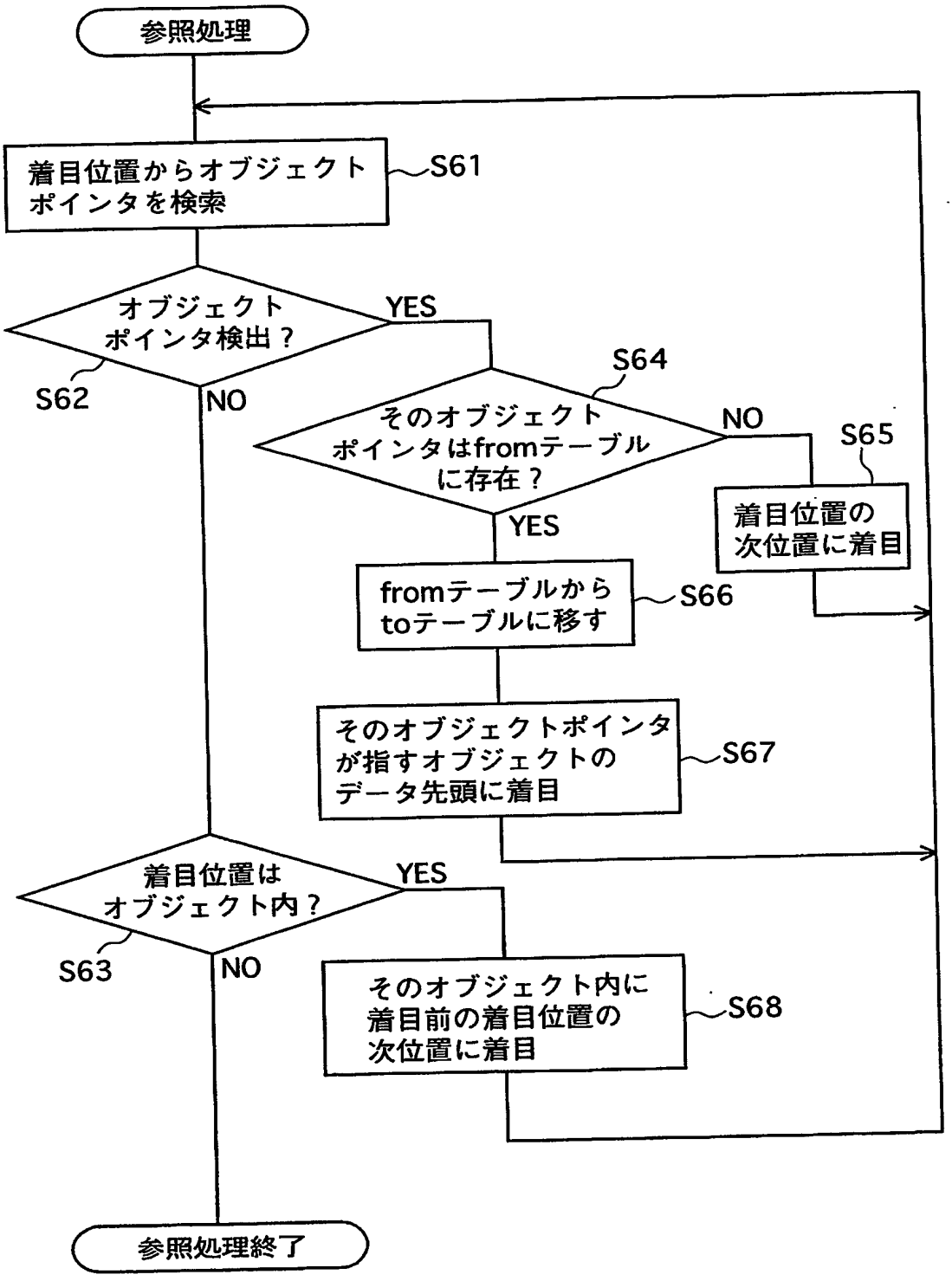
【図１０】


共有オブジェクト参照処理
共有オブジェクト管理情報の先頭に着目
S41
参照処理
S42
共有オブジェクト参照処理終了

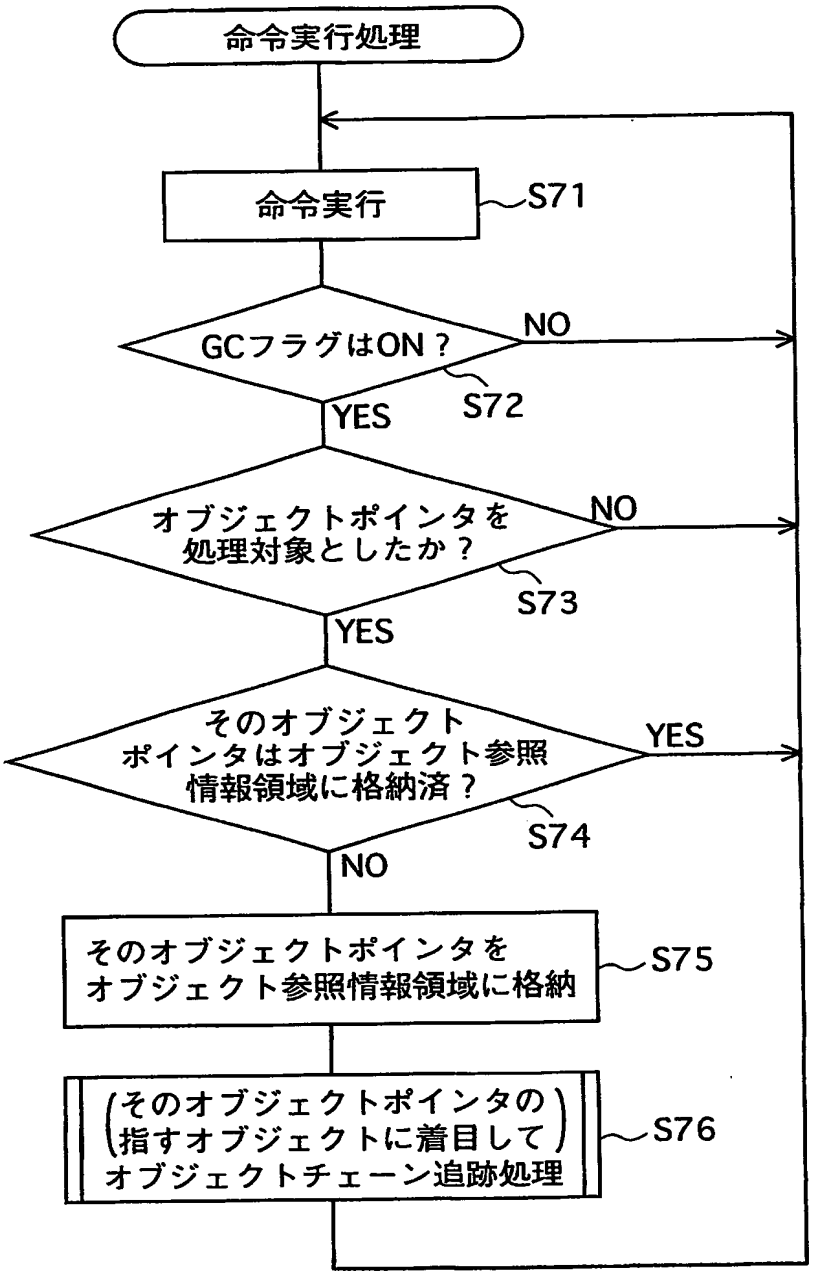
【図１１】


対象スレッド参照処理
GC開始時スタックポインタの位置に着目
S51
参照処理
S52
オブジェクト参照情報の先頭位置に着目
S53
参照処理
S54
対象スレッド参照処理終了

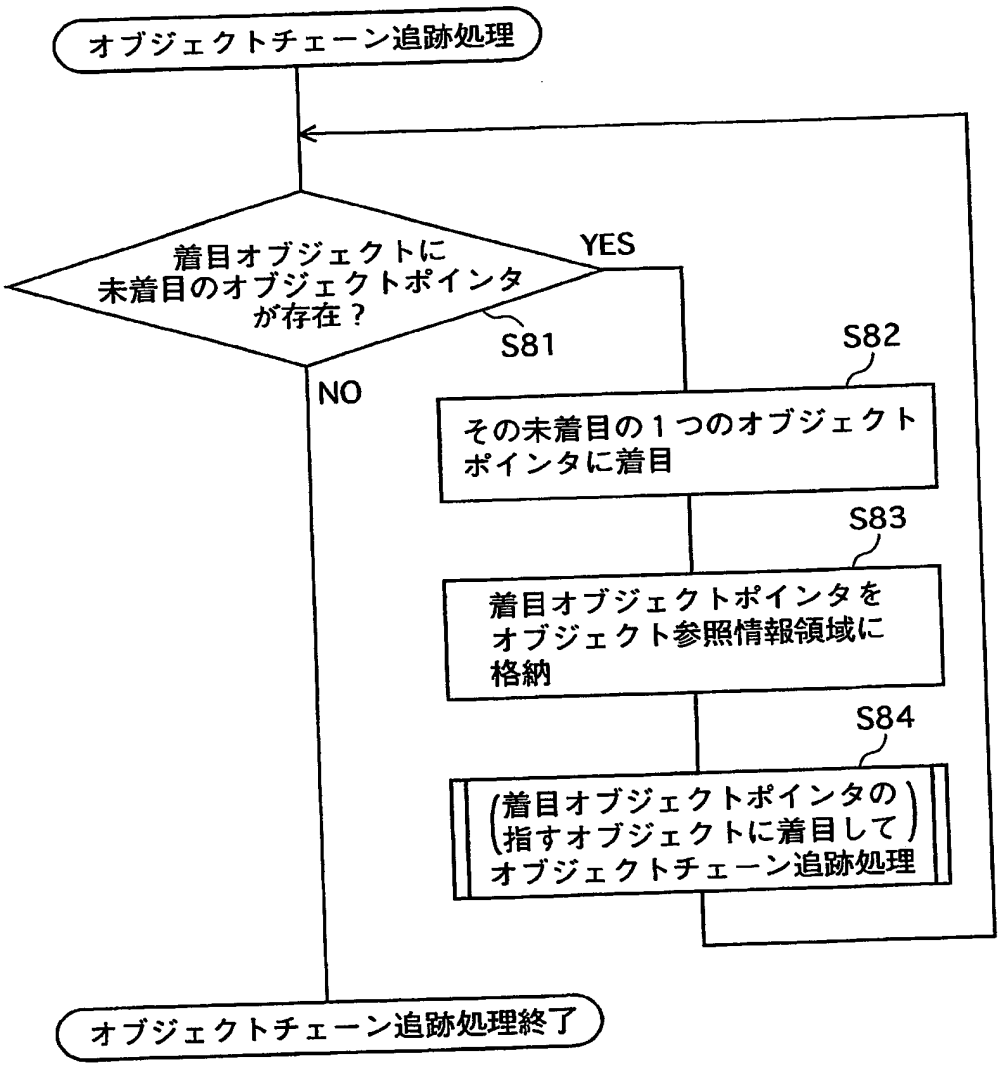
【図１２】


参照処理
着目位置からオブジェクトポインタを検索
S61
オブジェクトポインタ検出？
YES
S62
NO
S64
そのオブジェクトポインタはfromテーブルに存在？
NO
S65
着目位置の次位置に着目
YES
fromテーブルからtoテーブルに移す
S66
そのオブジェクトポインタが指すオブジェクトのデータ先頭に着目
S67
着目位置はオブジェクト内？
YES
S63
NO
そのオブジェクト内に着目前の着目位置の次位置に着目
S68
参照処理終了

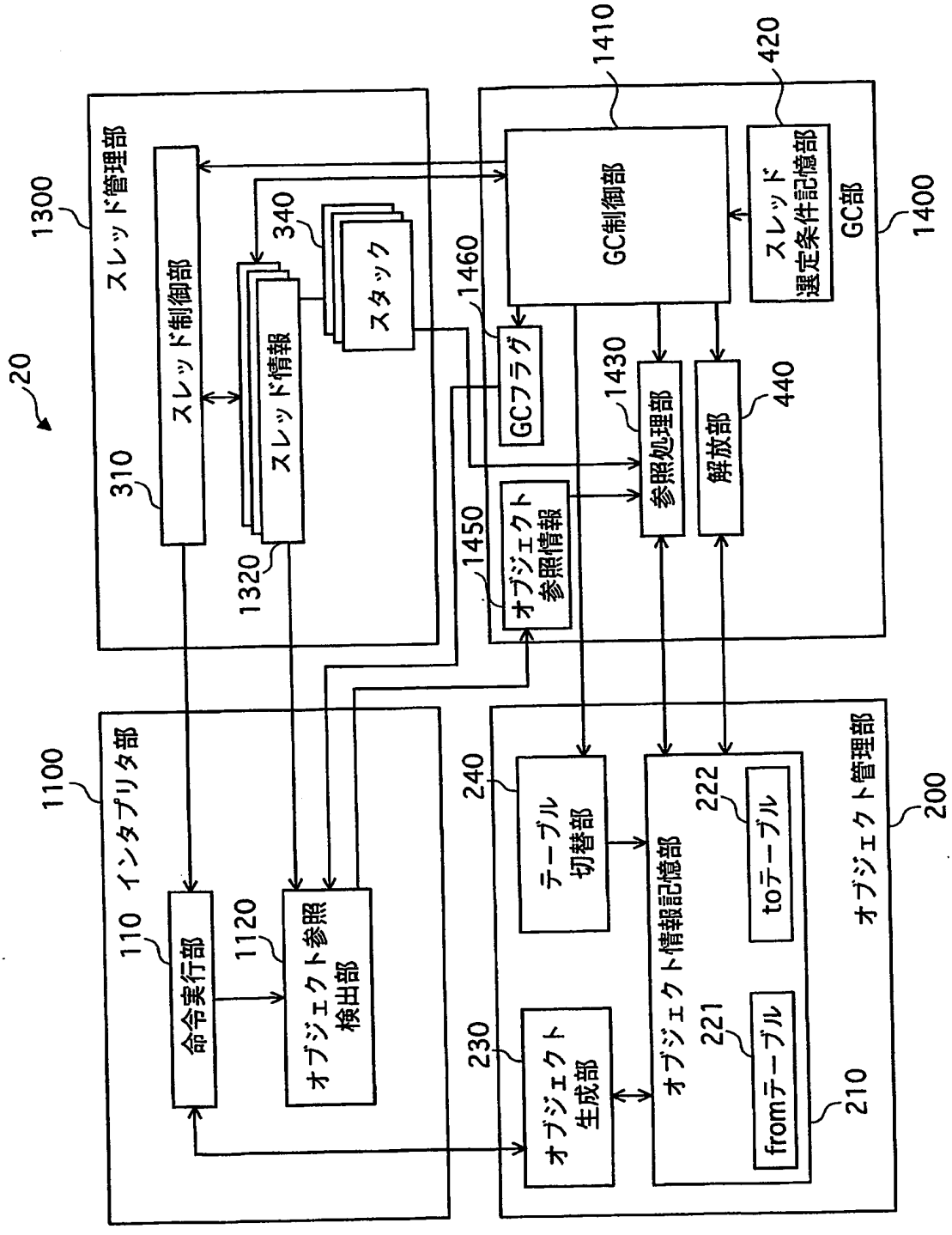
【図１３】


命令実行処理
命令実行
S71
GCフラグはON？
NO
S72
YES
オブジェクトポインタを
処理対象としたか？
NO
S73
YES
そのオブジェクト
ポインタはオブジェクト参照
情報領域に格納済？
YES
S74
NO
そのオブジェクトポインタを
オブジェクト参照情報領域に格納
S75
(そのオブジェクトポインタの
指すオブジェクトに着目して
オブジェクトチェーン追跡処理
S76

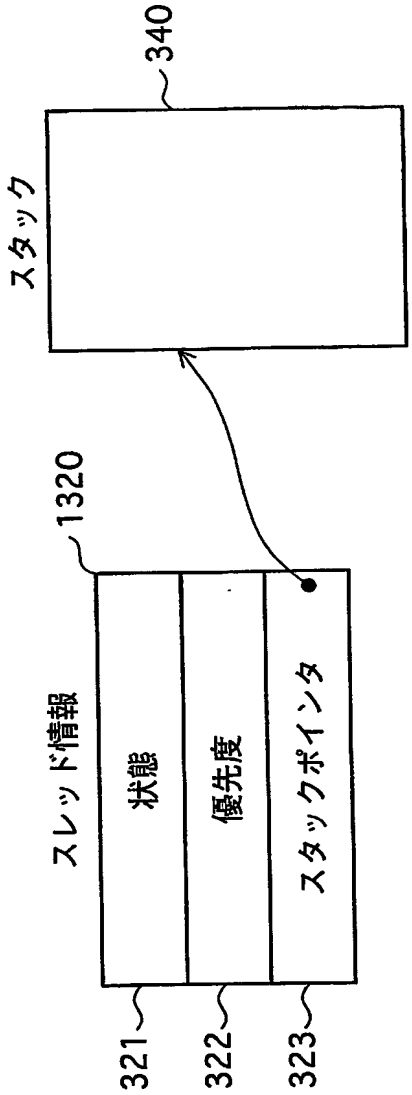
【図１４】


オブジェクトチェーン追跡処理
着目オブジェクトに
未着目のオブジェクトポインタ
が存在？
YES
NO
S81
S82
その未着目の１つのオブジェクト
ポインタに着目
S83
着目オブジェクトポインタを
オブジェクト参照情報領域に
格納
S84
着目オブジェクトポインタの
指すオブジェクトに着目して
オブジェクトチェーン追跡処理
オブジェクトチェーン追跡処理終了

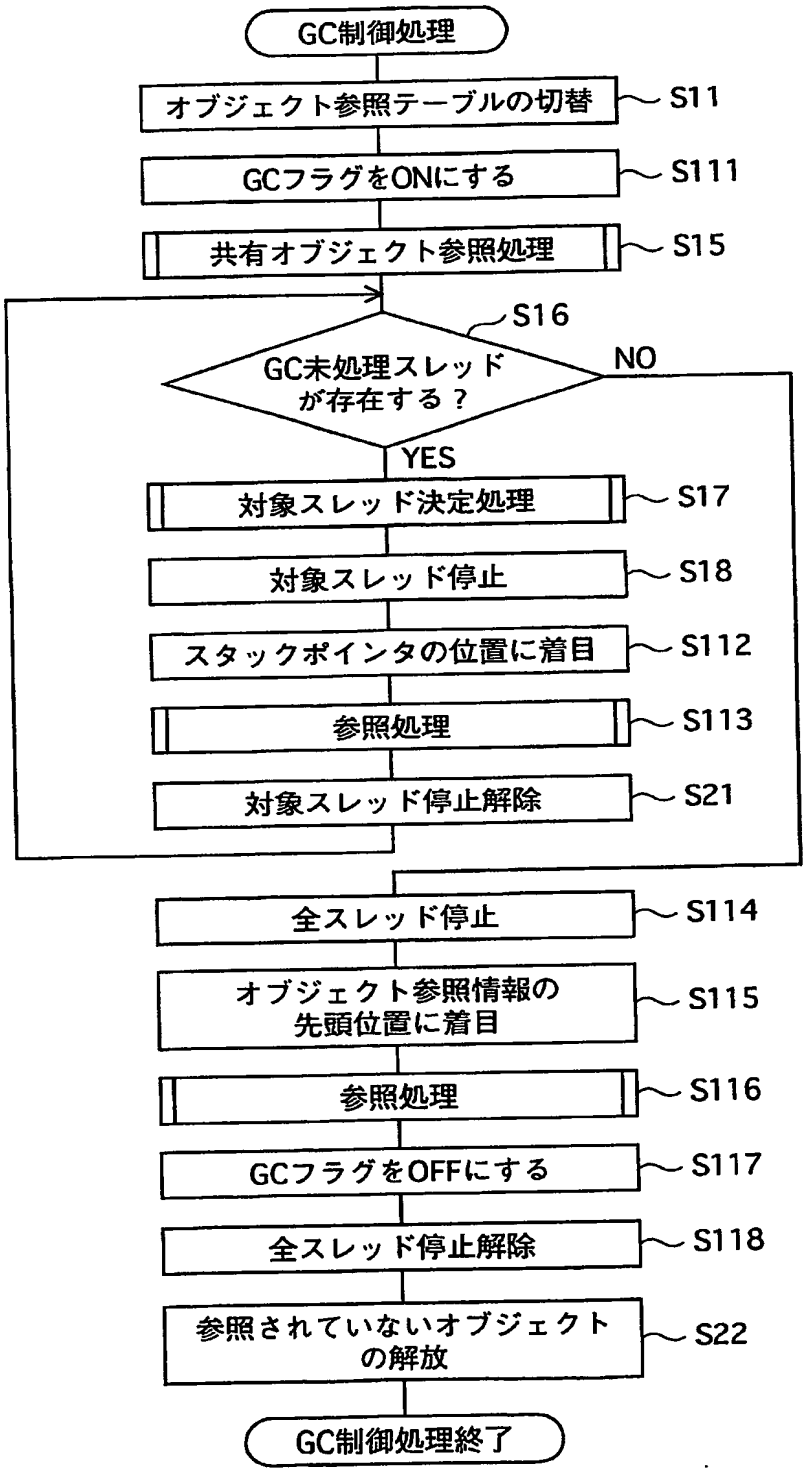
【図１５】


20
1300
スレッド管理部
310
スレッド制御部
340
スタック
スレッド情報
1320
1410
GC制御部
1460
GCフラグ
1430
参照処理部
1450
オブジェクト参照情報
420
スレッド選定条件記憶部
440
解放部
GC部
1400
1100
インタプリタ部
110
命令実行部
1120
オブジェクト参照検出部
240
テーブル切替部
222
toテーブル
オブジェクト情報記憶部
221
fromテーブル
210
230
オブジェクト生成部
オブジェクト管理部
200

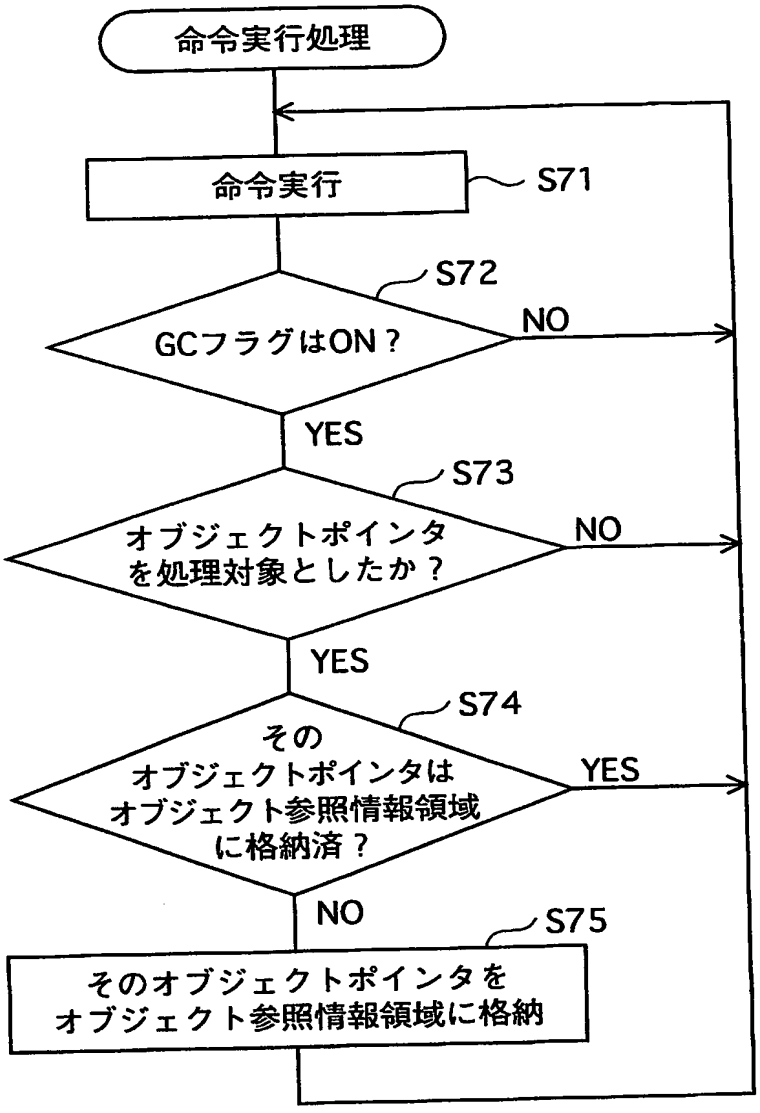
【図１６】



【図１７】


GC制御処理
オブジェクト参照テーブルの切替
S11
GCフラグをONにする
S111
共有オブジェクト参照処理
S15
S16
GC未処理スレッド
が存在する？
NO
YES
対象スレッド決定処理
S17
対象スレッド停止
S18
スタックポインタの位置に着目
S112
参照処理
S113
対象スレッド停止解除
S21
全スレッド停止
S114
オブジェクト参照情報の
先頭位置に着目
S115
参照処理
S116
GCフラグをOFFにする
S117
全スレッド停止解除
S118
参照されていないオブジェクト
の解放
S22
GC制御処理終了

【図１８】


命令実行処理
命令実行
S71
S72
GCフラグはON？
NO
YES
S73
オブジェクトポインタ
を処理対象としたか？
NO
YES
S74
その
オブジェクトポインタは
オブジェクト参照情報領域
に格納済？
YES
NO
S75
そのオブジェクトポインタを
オブジェクト参照情報領域に格納

【書類名】　要約書

【要約】

【課題】　ＡＰの全スレッド停止の期間を長期化させず、ＧＣに要するＣＰＵ時間の無駄な増加をある程度抑制するガーベジコレクション（ＧＣ）システムを提供する。

【解決手段】　ＧＣ制御部４１０は、スレッド選定条件記憶部４２０の内容に従って各スレッドを順次選択し、選択したスレッドについて、実行停止し、参照処理部４３０により、そのスレッドからオブジェクトポインタの参照を通じてアクセス可能なオブジェクトを検出してそのオブジェクトを非解放対象として管理する参照処理を実行して、そのスレッドの実行を再開する。また、インタプリタ部１００は、実行中スレッドによりオブジェクトポインタが処理対象にされたことを検知すればそのオブジェクトポインタを記録する。参照処理では更にその記録されたオブジェクトポインタから辿れるオブジェクトも非解放対象として管理する。解放部４４０は非解放対象以外のオブジェクトを解放する。

【選択図】　図１

特願２００３－１８７６９０

出 願 人 履 歴 情 報

識別番号 [０００００５８２１]

１．変更年月日 １９９０年 ８月２８日
[変更理由] 新規登録
住 所 大阪府門真市大字門真１００６番地
氏 名 松下電器産業株式会社